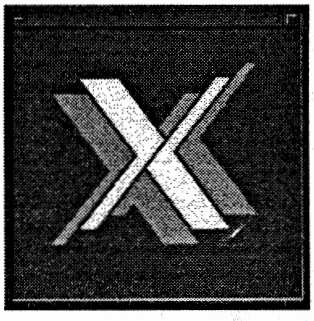
SourceGate™ II

Background Mode Debugger

for



User's Guide

Version 1.1.2

September 6, 1994

Huntsville Microsystems, Inc. © 1994

The contents of this document are believed to be accurate. However, Huntsville Microsystems, Inc. assumes no responsibility for the use of this document and issues no warranties, implied or otherwise, on the contents herein. Furthermore, Huntsville Microsystems, Inc. reserves the right to revise this publication and to make changes to the contents, without obligation to notify any persons of such revisions or changes.

Copyright © 1994 by Huntsville Microsystems, Inc.

SourceGate is a registered trademark of Huntsville Microsystems, Inc.
SUN logo, Solaris, and OpenWindows are trademarks of Sun Microsystems, Inc.
SPARC is a trademark of SPARC International Inc.
Hercules is a registered trademark of Hercules Computer Technology.
Pentium is a registered trademark of Intel Corporation.
Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation.

This document was created entirely with AmiPro, which is a trademark of Lotus Development Corporation.

TABLE OF CONTENTS

Chapter 1 - Getting Started 1

Checking Your Package 1

Disks 1

Documentation 1

Hardware 1

System Requirements 1

Hardware 1

Software 2

Memory 2

Disk Space 2

Installing The Windows Version Of The SourceGate II BMD 2

Installing The SUN Version Of The SourceGate II BMD 2

Installing The Motif Version Of The SourceGate II BMD 3

Starting The Windows Version Of The SourceGate II BMD 4

Starting The SourceGate II BMD From Program Manager 4

Starting The SourceGate II BMD From File Manager 4

Starting The SUN Version Of The SourceGate II BMD 4

Starting The Motif Version Of The SourceGate II BMD 5

Command Line Options 5

Environment Variables 7

BMD To Target System Connection 8

Chapter 2 - General Overview 9

Becoming Familiar With The SourceGate II BMD Window 9

Control Menu Boxes 9

Menu Bar 9

Title Bar 10

Restore Button 10

Maximize Button 10

Minimize Button 10

TABLE OF CONTENTS (Cont.)

Scroll Bars (Horizontal and Vertical) 10
Scroll Boxes (Horizontal and Vertical) 10
SourceGate II BMD Menu Bar 10
The 'File' Menu 10
Download Hex File 10
Upload Hex File 10
Execute 10
Record 10
Log 11
Print 11
Printer Setup 11
Exit SourceGate 11
The 'Edit' Menu 11
The 'Setup' Menu 11
The 'Display' Menu 11
Command Window 11
Watch 11
Registers 11
Memory 11
Stack Trace 11
Run Status 11
The 'Source' Menu 12
Run 12
Go From/To 12
Step 12
Stop 12
Reset Emulator 12
Symbols 12
Modules 12
Assemble 12

TABLE OF CONTENTS (Cont.)

VASM 12
Tracking Assembly 12
The 'Config' Menu 13
Source Code Interface 13
The 'Windows' Menu 13
Next Window 13
Previous Window 13
Cascade Windows 13
Close 13
Close All 13
Open Window List 13
The 'Help' Menu 13
On Context 13
On Window 13
Keyboard 14
Index 14
How Do I... 14
Help On Help 14
Search 14
About 14
Maximizing/Minimizing Windows 15
Window Manipulation 17
Changing To The Next Window 17
Changing To The Previous Window 18
Choosing From The Open Window List 18
Cascading All Open Windows 18
Closing The Current Active Window 18
Closing All Open Windows 18
Defining Your Printer 18
Working With Symbols 19
Using Help 19

TABLE OF CONTENTS (Cont.)

Context Sensitive Help 19
Active Window Help 20
Help Menu 20
On Context 20
On Window 20
Keyboard 20
Index 20
How Do I... 20
Help On Help 20
On Help 21
Search 21
About 21
Version 21
SourceGate II BMD Help Window 22
Help Window Buttons 22
Windows Version 22
UNIX Versions 23
Help Window Menus 23
Windows Version 23
UNIX Version 24
Windows Version Cross-References 24
Exiting The SourceGate II BMD 25
Automating The SourceGate II BMD 26
Accelerator Keys 27
Chapter 3 - The BMD Interface 28
Source Code Interface Configuration 28
Working With Object Files 29
Downloading An Object File 29
File Down Load Options Window Fields 30
'Target Offset' Field 30
'AUX Port Command' Field 31

TABLE OF CONTENTS (Cont.)

'Records To Load' Fields 31
'Port Selection' Fields 31
'Symbol Options' Fields 31
File Down Load Options Window Buttons 32
Uploading An Object File 33
File Up Load Options Window Fields 33
Records To Dump 33
Target Starting Address 33
Target Ending Address 33
File Up Load Options Window Buttons 34
Using The Command Window 35
Opening The Command Window 35
Entering Command Window Commands 36
Command Window Buttons 36
Using Symbols In The Command Window 37
Using Module Commands In The Command Window 38
:=A [module] Assembly Mode Command 38
:=C [module] Clear Symbol Table Command 38
:=D [module] Display Current Modes Command 39
:=M [module] Mixed Mode Command 39
:=S [module] Source Mode Command 39
Creating Automated Command Files From The Command Window 40
Available Command Window Commands 41
A (In-Line Assembly) Command 42
BM (Byte Mode) Command 43
C (Compare Memory Block) Command 44
CM (Show Status Of Control Board Memory) Command 46
D (Dump Memory Block) Command 47
DM (Data Memory Mode) Command 48
E/EN (Enter/Change Memory) Command 49
EXP (Examine/Change Peripheral) Command 50

TABLE OF CONTENTS (Cont.)

EXR (Examine/Change Peripheral Register) Command 51
F (Fill Memory Block) Command 52
G (Begin Real-Time Emulation) Command 53
HLD (Hex Listing Disabled) Command 54
HLE (Hex Listing Enabled) Command 55
I/IN (Input) Command 56
L (List Disassembled Code) Command 57
LM (Long Word Display Mode) Command 58
M (Move Memory Block) Command 59
O/ON (Output) Command 60
PM (Program Memory Mode) Command 61
RD (Read Object File Into Memory) Command 62
RP (Reset Processor, Continue Emulation) Command 64
RS (Reset Processor, Stop Emulation) Command 65
SM (Supervisor Memory Mode) Command 66
SP (Stop Emulation) Command 67
SR/SRN (Search Block Of Memory) Command 68
SS (Single Step) Command 70
ST (Show Emulator Status) Command 73
UM (User Memory Mode) Command 74
VASM (View Assembly) Command 75
VER (Display Firmware Version) Command 76
W (Write Memory Contents To File) Command 77
WM (Word Display Mode) Command 79
X (Examine/Change Registers) Command 80
Working With The Watch Window 82
Opening The Watch Window 82
Adding A Variable To The Watch Window 83
Entering Watch Variables Manually From The Watch Window 83

TABLE OF CONTENTS (Cont.)

Using The Watch Entry Window To Enter Watch Variables 85
Adding A Watch Variable Manually 85
Adding A Watch Variable Using Buttons 87
Deleting A Variable From The Watch Window 89
Saving A Watch Window 89
Restoring A Saved Watch Window 89
Watch Window Buttons 89
Watch Window Examples 90
Watching A Structure 90
Watching An Element Of A Structure 91
Watching A One-Dimensional Array 91
Watching A Multi-Dimensional Array 91
Watching Pointers 92
Using Overrides In The Watch Window 92
Working With The Register Window 93
Opening The Register Window 94
Modifying Register Values 94
Working With The Memory Window 95
Opening The Memory Window 95
Modifying Memory Values 96
Comparing Memory Blocks 96
Memory Compare Window Buttons 96
Memory Output Window Buttons 97
Filling Memory With A Constant 98
Memory Fill Window Buttons 98
Viewing A Specific Address In Memory 99
Memory Goto Window Buttons 99
Reading/Writing A Specific Memory Location 100
Memory I/O Window Buttons 100
Moving Memory Blocks 101
Memory Move Window Buttons 101

TABLE OF CONTENTS (Cont.)

Searching Memory Blocks 102
Memory Search Window Buttons 102
Memory Search Output Window Buttons 103
Specifying The Display Format 104
Memory Display Format Window Buttons 104
Memory Window Buttons 105
Working With The Stack Trace Window 106
Opening The Stack Trace Window 106
Stack Trace Window Contents 106
Working With The Run Status Window 108
Opening The Run Status Window 108
Run Status Window Fields 108
Run Status 'Status' Field 108
Run Status Trace Field 109
Run Status Pass Count Fields 109
Run Status Timer Field 109
Run Status Window Buttons 109
Chapter 4 - The Source Code Interface 110
Using The Code Window 110
Opening A Code Window 111
Code Window Format 112
Setting Code Window Breakpoints 113
Deleting Code Window Breakpoints 113
Listing Code Window Breakpoints 115
Breakpoint List/Delete Breakpoint Window Buttons 115
Finding Subsequent Code Window Breakpoints 116
Finding The Current Program Counter While In A Code Window 116
Running To A Highlighted Code Window Line 116
Single Stepping In A Code Window 116
Searching For A Code Window String 117
Viewing Code Window Assembly 118

TABLE OF CONTENTS (Cont.)

Viewing Code Window Symbols 119
Selecting Other Modules From A Code Window 120
Using The VASM Window 121
Using The Tracking Assembly Window 122
Using The Single Step Window 123
Single Step Window Fields 123
Address Field 123
Current Instruction Field 123
Step Continuously Check Box 123
Step Over Subroutines Check Box 123
Quiet Mode Check Box 123
Step Count Field 124
Step Until Field 124
Single Step Window Buttons 125
Using The Go From/To Window 126
Go From/To Window Fields 126
Go From Address Field 126
Go To Breakpoints Fields 126
Go From/To Window Buttons 127
Appendix A Error Messages 129
Appendix B Using Compilers and Assemblers 134

1 Getting Started

Checking Your Package

This section lists the disks, documentation, and hardware included in your SourceGate II Background Mode Debugger (BMD) emulation package. Take time now to check the contents of your package. If your package is not complete, contact HMI. **Note that the window examples shown in this document were captured under Microsoft Windows and may not reflect the exact "look" of the windows shown in your particular environment. However, the window contents, and the operations that are performed, should be identical.**

Disks

Your SourceGate II BMD package should contain one 3.5" disk (two disks for UNIX versions) labeled 'SourceGate II BMD'.

Documentation

Your package should contain the following documentation:

- *How to Use This Manual*
- *SourceGate II Background Mode Debugger User's Guide*
- *Application Notes*

Hardware

Your package should contain the BMD ordered along with a cable that has a 25-pin female connector on one end and a 8/10 pin Berg connector on the other end. The 25-pin connector should be connected to your computer's serial interface using a straight-thru cable. The Berg connector should be connected to either an 8 or 10 pin Berg connector on your target system.

System Requirements

This section describes the hardware, software, memory, and disk space requirements for using the SourceGate II BMD on a stand-alone computer or on a network.

Hardware

The SourceGate II BMD requires the following hardware:

- *A 286, 386, 486, or Pentium™ based computer certified for use with Windows 3.1, a SUN SPARC computer, or a HP 9000 Series 700 computer*
- *A mouse is required for UNIX versions and is recommended for the Windows version*
- *One 1.44MB 3.5" disk drive*
- *One RS232 serial port*
- *Hard disk drive with at least 4MB of free disk space*

Software

You must have Microsoft Windows Version 3.1 or higher and DOS Version 3.1 or higher installed on your computer to use the Windows version of the SourceGate II BMD. You must have SUN OS Version 4.1.x and OpenWindows Version 3.0 or higher or Solaris Version 2.x installed on your computer to use the SUN version of the SourceGate II BMD. You must have HP-UX Version 8.x running the OSF/Motif GUI installed on your computer to use the Motif version of the SourceGate II BMD.

Memory

The Windows version of the SourceGate II BMD requires a minimum of 2MB of extended memory with 4MB recommended.

Disk Space

The SourceGate II BMD requires a minimum of 4MB of hard disk space.

Installing The Windows Version Of The SourceGate II BMD

To install the Windows version of the SourceGate II BMD, place the disk labeled "SourceGate II BMD" in your 1.44MB 3.5" disk drive, open the Windows File Manager, make the drive where the floppy is located to be the active drive, and run the "SETUP.EXE" program. You should then follow the instructions in the setup program in order to install the SourceGate II BMD files correctly. A Program Group will be created with the SourceGate II BMD icon shown in this group.

Installing The SUN Version Of The SourceGate II BMD

To install the SUN version of the SourceGate II BMD, perform the following steps:

cd <to the directory where you want the "hmi/sg2bmd/xxxxx" directory to reside, where xxxxx is the processor being installed (68332, 68340, etc.)>
<Insert Disk 1>
tar xvf /dev/<floppy name>
<Eject Disk 1 and insert Disk 2>
tar xvf /dev/<floppy name>
<Eject Disk 2>
cd hmi/sg2bmd/xxxxx (where xxxxx is the processor that was installed)
./sg2install

After the above steps have been completed, copy the sg2APP.color (for color systems) or sg2APP.mono (for monochrome systems) file to the /usr/openwin/lib/app-defaults directory into a file called 'sg2'. Some examples follow:

For color systems, type:

cp sg2APP.color /usr/openwin/lib/app-defaults/sg2

For monochrome systems, type:

cp sg2APP.mono /usr/openwin/lib/app-defaults/sg2

If you are not using Sun OpenWindows, you may need to copy the file to the X-Windows app-defaults directory. For example, if this directory is named /usr/X11/lib/app-defaults/sg2, then the following command needs to be performed :

cp sg2APP.color(or mono) /usr/X11/lib/app-defaults/sg2

Be certain that the LD_LIBRARY_PATH environment variable is correct. You can check it with the command 'printenv'. If it is wrong, then you will need to modify it as follows:

setenv LD_LIBRARY_PATH /usr/openwin/lib (or whatever the correct path is)

The file sg2.frl is a resource file. It must be in the current directory or in a directory in the PATH. The files <bmdname>.sg and <bmdname>.sg2 are BMD specific configuration files and <bmdname>.csc is a help file. All of these files must reside in the current directory or in a directory in your PATH environment variable.

Printing support is only available if you set the environment variable XVTPATH to the directory 'print' that is supplied on the SG2 disks. If you don't set this environment variable to this directory, any print operation will respond with an error message and exit. **Note that this version of SourceGate II only allows printing to a file.**

Installing The Motif Version Of The SourceGate II BMD

To install the Motif version of the SourceGate II BMD, perform the following steps:

cd <to the directory where you want the "hmi/sg2bmd/xxxxx" directory to reside, where xxxxx is the processor being installed (68332, 68340, etc.)>
<Insert Disk 1>
tar xvf /dev/<floppy name>
<Eject Disk 1 and insert Disk 2>
tar xvf /dev/<floppy name>
<Eject Disk 2>
cd hmi/sg2bmd/xxxxx (where xxxxx is the processor that was installed)
./sg2install

The file sg2.uid is a resource file. It must be in the current directory or in a directory in the UIDPATH. For example, if you wish to use the directory /usr/local/lib/sg2bmd to hold the UID file for the SourceGate II BMD (and for any other applications), you might include this line in your .cshrc file:

```
setenv UIDPATH /usr/local/lib/sg2bmd/%U
```

or these lines in your .profile:

```
UIDPATH=/usr/local/lib/sg2bmd/%U
export UIDPATH
```

The files <bmdname>.sg and <bmdname>.sg2 are BMD specific configuration files and <bmdname>.csc is a help file. All of these files must reside in the current directory or in a directory in your PATH environment variable.

Starting The Windows Version Of The SourceGate II BMD

The Windows version of the SourceGate II BMD can be started from Windows in several ways. There are also environment variables and command line options that can be defined that influence the way the SourceGate II BMD will behave. First, however, you need to connect a background mode equipped target system to your computer using the supplied cable. See the section ***BMD To Target System Connection*** for information on how to do this.

Starting The SourceGate II BMD From Program Manager

To start the SourceGate II BMD from Program Manager, double-click on the SourceGate II BMD icon located in the HMI program group. **This is the recommended method of starting the SourceGate II BMD.** You can also choose "File/Run" from the Program Manager menu bar and enter the path of the SG2BMD.EXE file, along with any command line options, and then choose OK. For additional information on the available command line options, see the section ***Command Line Options***.

Starting The SourceGate II BMD From File Manager

To start the SourceGate II BMD from File Manager, go to the directory where the SourceGate II BMD files are located, highlight the SG2BMD.EXE file, and choose OK.

Starting The SUN Version Of The SourceGate II BMD

To start the SUN version of the SourceGate II BMD, you will first need to connect a background mode equipped target system to your computer using the supplied cable. See the section ***BMD To Target System Connection*** for information on how to do this. After the BMD to target system connection has been made, change to the directory where the SourceGate II BMD resides and enter the following:

```
sg2bmd
```

This will start the SourceGate II BMD and it will try to communicate with the target system on /dev/tty00 running at 9600 baud. If you wish to start the SourceGate II BMD using other optional parameters, they should be entered as follows:

sg2bmd <Command Line Options>

For additional information on the available command line options, see the section ***Command Line Options***.

Starting The Motif Version Of The SourceGate II BMD

To start the Motif version of the SourceGate II BMD, you will first need to connect a background mode equipped target system to your computer using the supplied cable. See the section ***BMD To Target System Connection*** for information on how to do this. After the BMD to target system connection has been made, change to the directory where the SourceGate II BMD resides and enter the following:

sg2bmd

This will start the SourceGate II BMD and it will try to communicate with the target system on /dev/tty00 running at 9600 baud. If you wish to start the SourceGate II BMD using other optional parameters, they should be entered as follows:

sg2bmd <options>

Command Line Options

The following command line options can be entered for the SourceGate II BMD program. To add these parameters for the Windows version of the SourceGate II BMD icon, highlight the SourceGate II BMD icon by clicking on it once. Then choose 'File/Properties' and entered the desired command line options on the command line after the SG2BMD.EXE filename. Be sure and leave a space between each option. After you are done adding options, choose OK to save the new properties. The next time the SourceGate II BMD is started by double-clicking on the SourceGate II BMD icon, the new options will be in effect. If you are using one of the UNIX versions of the BMD, consult your UNIX documentation for information on how to define a script for the SUN or Motif versions of the SourceGate II BMD that will implement these command line options.

The available command line options are as follows:

-<Baud>
Select communications port baud rate. Allowable selections are: 9600 (default), 19200, and 57600. If you specified a specific baud rate during the install process, it will be the default. **Note: Windows does not support baud rates higher than 57600 baud. Most UNIX machines will not support baud rates greater than 38400.**

-<Port>
Select communications port. The Windows version of the SourceGate II BMD automatically uses COM1 as its default communications port, unless COM1 is not available, in which case it will use COM2. The -port option will override the default SourceGate II BMD port selection and cause the SourceGate II BMD to use only the port named. Allowable options are: COM1, COM2, COM3, COM4 and AUX. If you are using one of the UNIX versions of the BMD, consult your UNIX documentation for information on valid serial port devices for your particular computer.

-L<LineSymbol>
In order to enable source-level debugging, the SourceGate II BMD must know the relationship between the lines in the source file on disk, and the assembly language code in the target system. This relationship is defined through the use of Line Number Symbols.

The default format for the line number symbols recognized by the SourceGate II BMD is "L_%u". This is a scanf-formatted string indicating that line number symbols are those symbols which consist of one capital 'L' character, followed by an underscore character and an unsigned decimal number. The unsigned decimal number is the line number. The value associated with the symbol is the starting address of the assembly-language instruction generated by that line of source code.

If, for example, your compiler produced line number symbols of the format LL_%u, then the-LLL_%u option should be used on the command line which initiates the SourceGate II BMD. In this example, the -L tells the SourceGate II BMD that the line number format is LL_%u. This command line switch allows source-level debugging using any compiler that produces line number symbols.

-E<SrcExt>
This switch defines the source code extent. The default extent is ".C". When source level or mixed mode debugging is enabled, the SourceGate II BMD will look for source modules with the defined extent.

-B<CommandFile>
This option defines a command (.CMD) file to be automatically executed upon entering the SourceGate II BMD.

<HexFile>
This option establishes the default program filename for RD and W commands.

Examples:

sg2bmd	= Start the SourceGate II BMD with default options
sg2bmd -19200	= Set 19200 Baud
sg2bmd -COM2	= Use COM2
sg2bmd -LLL_%u	= Set Line Number Symbol format to "LL_%u"
sg2bmd -E68K	= Set extent default to ".68K"
sg2bmd -BTEST	= Define "TEST.CMD" as command file to be run when the SourceGate II BMD is invoked
sg2bmd DEMO.OUT	= Establish "DEMO.OUT" as default filename for RD and W commands

If all of the above options were used, the command line would look like the following:

sg2bmd -19200 -COM2 -LLL_%u -E68K -BTEST DEMO.OUT

Environment Variables

Several environment variables are used by the Windows version of the SourceGate II BMD. These environment variables can be defined by using the DOS "SET" command. If you are using one of the UNIX versions of the BMD, consult your UNIX documentation for information on how to define environment variables.

The available environment variables are defined as follows:

HMI_BAUD:
Environment variable for the baud rate of the serial link between the BMD and the target system. Options are 9600, 19200, and 57600. **Note: Windows does not support baud rates higher than 57600 baud. Most UNIX machines will not support baud rates greater than 38400.**

HMI_LINE:
Environment variable for line number format.

HMI_PORT:
Environment variable for communications port being used to communicate between the BMD and the target system. Options are COM1, COM2, COM3, COM4, and AUX for the Windows version. If you are using one of the UNIX versions of the BMD, consult your UNIX documentation for information on valid serial port devices for your particular computer.

HMI_STARTUP:
Environment variable for setting the command file that gets executed when the SourceGate II BMD is invoked. This is handy for automating the loading of a file when the SourceGate II BMD is run.

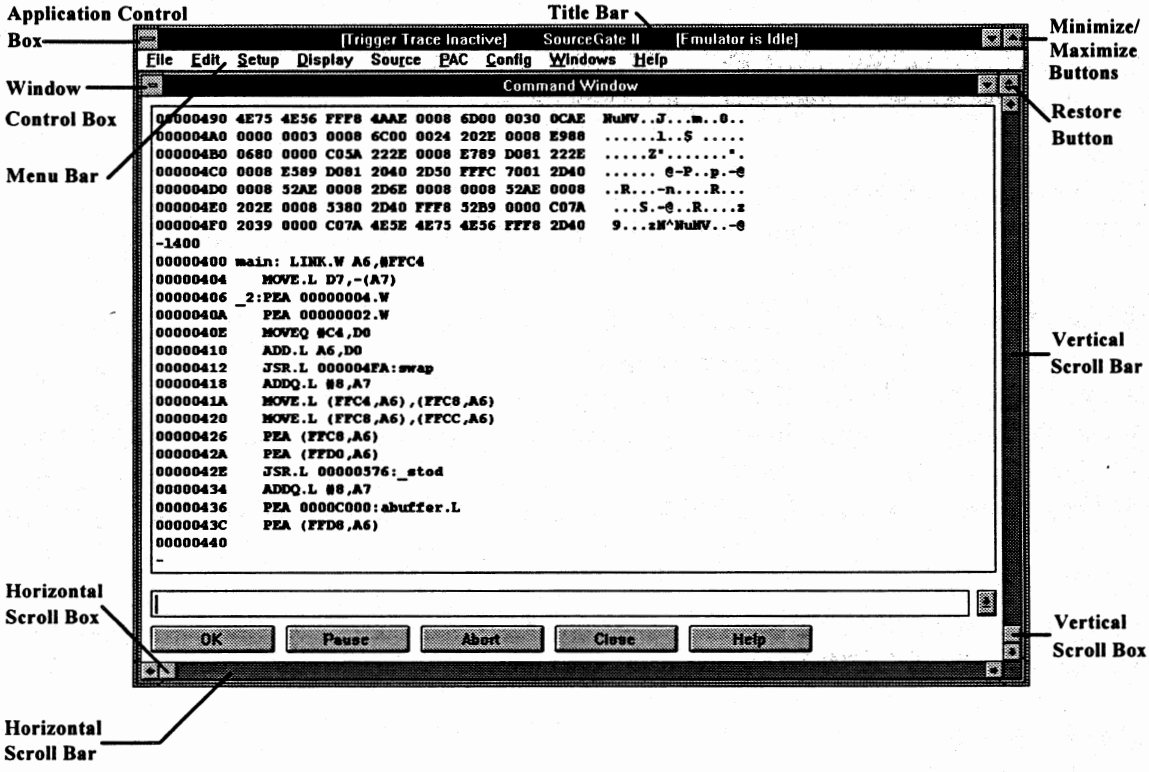
BMD To Target System Connection

Before starting the SourceGate II BMD, you must connect a serial cable from a serial port of the computer to the 25-pin female connector on the HMI supplied cable. This should be a "straight-thru" or "modem" type cable. Then, connect the 8 or 10 pin Berg connector to the Berg connector on your target system.

2 General Overview

Becoming Familiar With The SourceGate II BMD Window

Take a moment to review the SourceGate II BMD window on this page. Become familiar with the various parts of the SourceGate II BMD window so you can work quickly and efficiently. **Note that the window examples shown in this document were captured under Microsoft Windows and may not reflect the exact "look" of the windows shown in your particular environment. However, the window contents, and the operations that are performed, should be identical.**



An explanation of each of these window controls is given as follows:

Control Menu Boxes

Displays a menu with commands that allow you to restore, move, size, minimize, maximize, or close the application/active window. The application control menu box also allows you to switch between windows or open the control panel window.

Menu Bar

Lists the main menus in the SourceGate II BMD.

Title Bar
Displays the name of the application (SourceGate II BMD) and shows the status of the BMD (Idle or Running).

Restore Button
Restores the SourceGate II BMD application or window to the size and position it occupied before you maximized or minimized it.

Maximize Button
Enlarges the SourceGate II BMD application or window to fill the entire screen.

Minimize Button
Reduces the SourceGate II BMD application or window to an icon at the bottom of the screen.

Scroll Bars (Horizontal and Vertical)
Controls the portion of window text that appears on the screen.

Scroll Boxes (Horizontal and Vertical)
Indicates where the screen display is located compared to the full width of the window.

SourceGate II BMD Menu Bar
The menu bar for the SourceGate II BMD provides you a way to pull up the main menus that can be used with the SourceGate II BMD. An explanation of each of these menus follows. Note that the menu bar choices might change depending on the current active window. The following menu choices are available when the main SourceGate II BMD window is active. **If a menu item is greyed out, it has not been implemented in this version of the SourceGate II BMD.**

The 'File' Menu
The main function of the 'File' menu is getting information into and out of the SourceGate II BMD. The following is a brief explanation of the options available in this menu:

Download Hex File
Download an object file into memory.

Upload Hex File
Upload memory into an object file that can be used to burn PROMs.

Execute
Execute a pre-recorded command file.

Record
Record a command file.

Log
Begin logging all keyboard input and screen output to an ascii file that can be reviewed at a later date. This enables you to obtain a "log" file of what you have done.

Print
Print the contents of the active window to a file or a printer. **This menu item will be greyed out if the active window does not support printing operations.**

Printer Setup
Setup printer parameters.

Exit SourceGate
Exit the SourceGate II BMD program.

The 'Edit' Menu
The 'Edit' menu will allow you to cut and paste information from/to the SourceGate II BMD windows. This menu is not implemented in this version of the SourceGate II BMD.

The 'Setup' Menu
The 'Setup' menu is used to "setup" the event, breakpoint, trace trigger, interval timer, and external output windows. This menu is not implemented in this version of the SourceGate II BMD.

The 'Display' Menu
The 'Display' menu allows the user to choose certain windows for display. The following is a brief explanation of the options available in this menu:

Command Window
Open the Command Window and make it the active window.

Watch
Open the Watch Window and make it the active window.

Registers
Open the Register Window and make it the active window.

Memory
Open the Memory Window and make it the active window.

Stack Trace
Open the Stack Trace Window and make it the active window.

Run Status
Open the Run Status Window and make it the active window.

The 'Source' Menu

The main function of the 'Source' menu is to provide an interface when dealing with source code information. The following is a brief explanation of the options available in this menu:

Run

Begin emulation from the current program counter location. This item will be greyed out if emulation is in progress.

Go From/To

Open the 'Go From/To' window which allows the user to begin emulation from a specified address and also allows temporary breakpoints to be set.

Step

Open the 'Single Step' window which allows the user to single step through instructions.

Stop

Stop emulation and update all window values with the latest information. This item will be greyed out if emulation is not in progress.

Reset Emulator

Reset the target system's processor by passing the RESET signal to the target system.

Symbols

Open the 'Symbol Selection' window which allows you to view and select symbols for a particular module.

Modules

Open the 'Module Selection' window which allows you to view and select a source module to work with.

Assemble

Open an assembly window for the current module. You can then work at the assembly level.

VASM

Open an assembly window beginning at the address defined by the user.

Tracking Assembly

Invoke an assembly window which will remain on the screen and will track the source window that currently has the focus by highlighting the assembly lines that correspond to the source line highlighted in the active module. If this window is activated when no source modules are open, it will show assembly code beginning at the current program counter location.

The 'Config' Menu

The 'Config' menu deals with setting up BMD options. The following is a brief explanation of the options available in this menu:

Source Code Interface
Define the relationship between the SourceGate II BMD and the compiler being used.

The 'Windows' Menu

The 'Window' menu allows you to manipulate the SourceGate II BMD windows. The following is a brief explanation of the options available in this menu:

Next Window
Open the next window in the open window list of previously opened windows.

Previous Window
Make the window that was active prior to the current window to be the active window.

Cascade Windows
Cascades all open windows.

Close
Close the open window that currently has focus.

Close All
Close all open windows.

Open Window List
Not an actual option, this list will show all windows that are currently open. You can make a window in this list to be the active window by clicking on it. If no windows are open, "No Windows" will be shown.

The 'Help' Menu

The 'Help' menu provides help on using the SourceGate II BMD. The following is a brief explanation of the options available in this menu:

On Context
Provides help on the current active window. This is the same as 'On Window'. Future versions of the SourceGate II BMD may use this option to provide help on particular fields instead of at the window level.

On Window
Provides help on the current active window. This is the same as 'On Context'.

Keyboard
Provides help on commands that can be executed from the keyboard.

Index
Shows an index of the topics available in the SourceGate II BMD Help System.

How Do I...
Provides help for some of the more commonly asked questions.

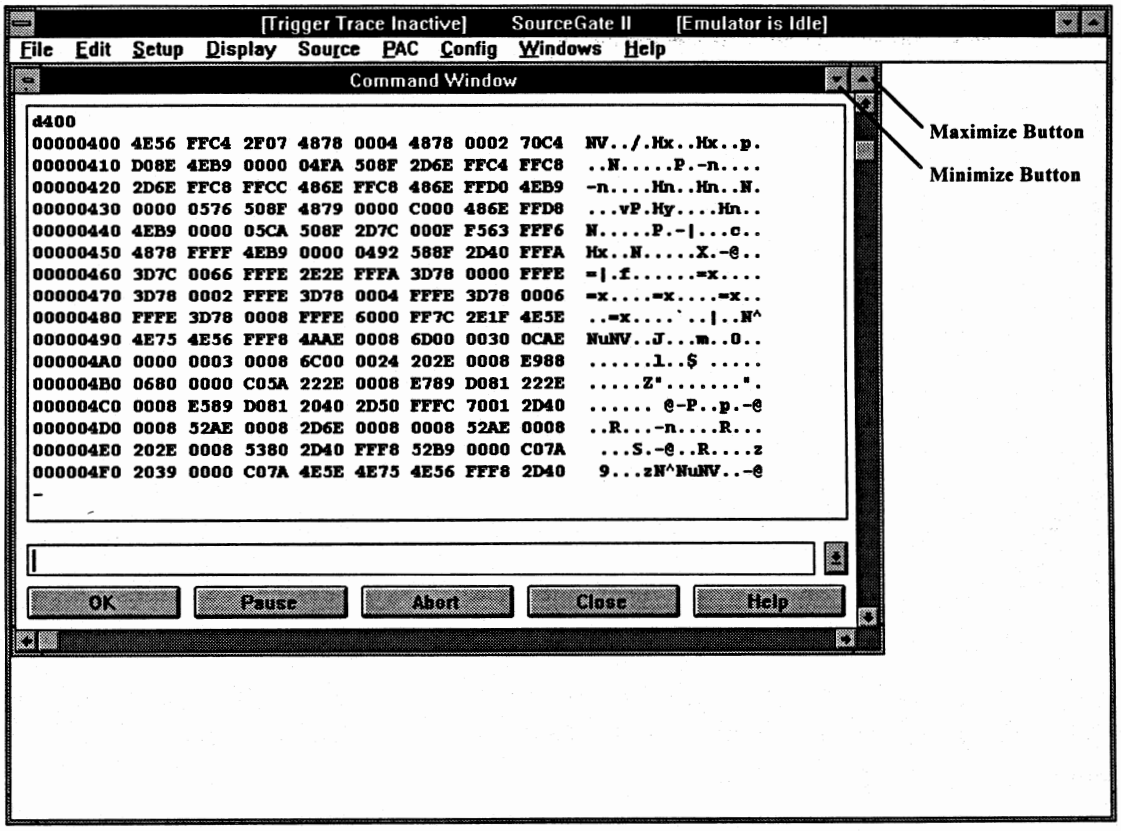
Help On Help
Provides help on how to use the help system.

Search
Allows you to search for a particular topic by using keywords.

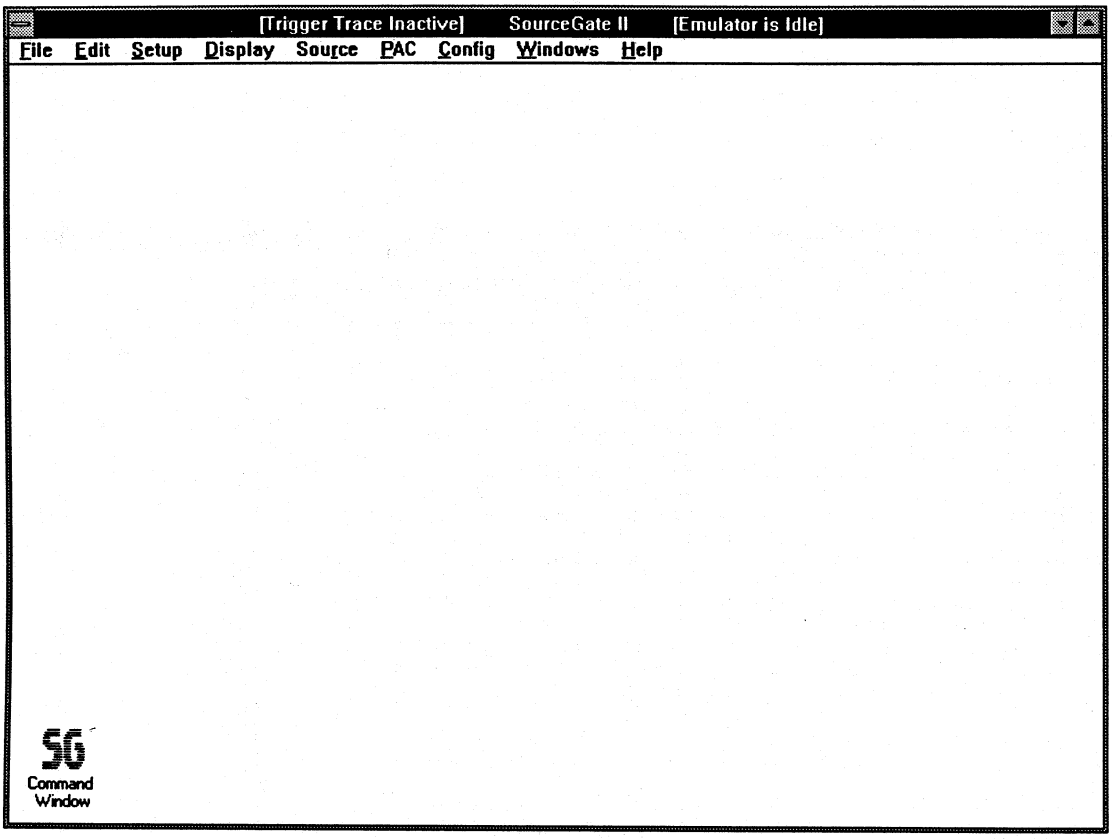
About
Shows information about the version of the help files.

Maximizing/Minimizing Windows

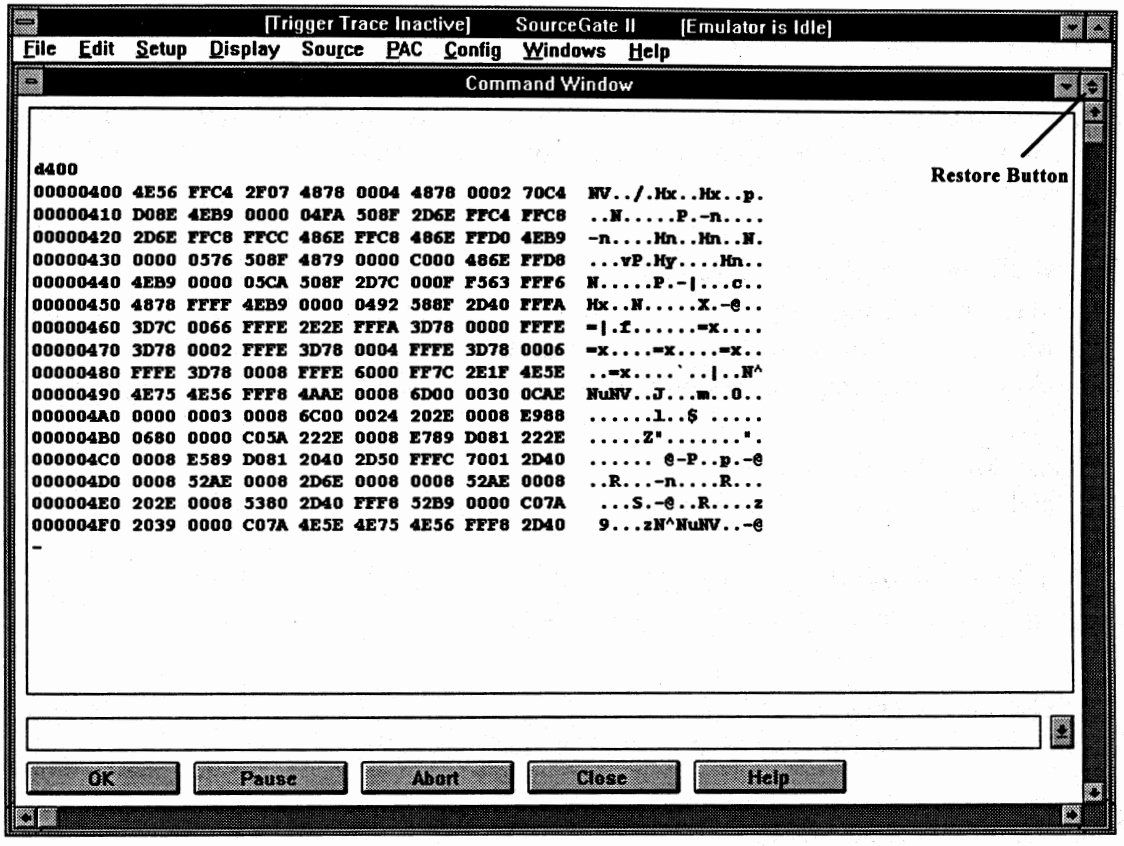
By using the maximize and minimize buttons, you can make the SourceGate II BMD windows to be icons or windows of a certain size. An example may help. Consider the main SourceGate II BMD window shown below:



Notice that the Command window has a minimize and a maximize button. If the minimize button is pressed, the Command window will be reduced to an icon as shown in the following example:


[Trigger Trace Inactive] SourceGate II [Emulator is Idle]
File Edit Setup Display Source PAC Config Windows Help
SG
Command
Window

You can double-click on the icon to make the window active again. If the maximize button is then pressed, the Command window will take on its largest size as shown in the following example:



Pressing the restore button will restore the Command window back to its original size.

Windows can also be custom sized by clicking and dragging the window. For more information on how to do this, consult the Windows manual or the documentation for your particular UNIX platform.

Window Manipulation

Several windows can be shown at the same time. However, you may wish one window to "have focus" or be the top-most window. The following sections describe how to work with multiple windows.

Changing To The Next Window

Once several windows have been opened, you may choose the 'Windows' menu command. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make the window listed below the checked one to be the active window, choose ***Next Window***.

Changing To The Previous Window

Once several windows have been opened, you may choose the 'Windows' menu command. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make the window listed above the checked one to be the active window, choose ***Previous Window.***

Choosing From The Open Window List

Once several windows have been opened, you may choose the 'Windows' menu command. You will then be shown a listing of the windows that are open. A check mark will be next to the active window. To make another window in the list to be the active window, simply click on it.

Cascading All Open Windows

To cascade all open windows, choose ***Cascade Windows.***

Closing The Current Active Window

To close the current active window, choose ***Close.***

Closing All Open Windows

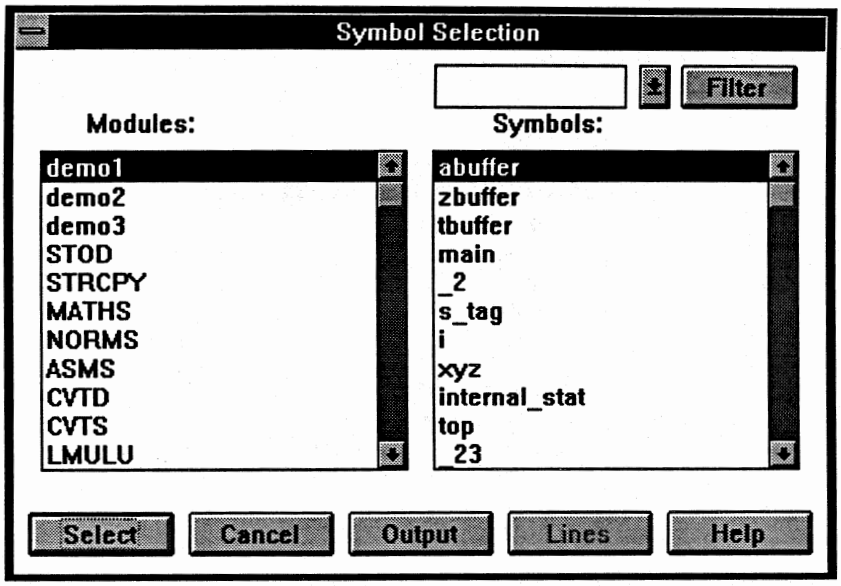
To close all windows that are currently open, choose ***Close All.***

Defining Your Printer

You can define the printer you wish to use with the SourceGate II BMD by choosing 'File/Printer Setup'. You will be shown the current printer and you can then change to another printer, if you wish, provided you have defined an alternate printer using the Windows 'Setup' group. For additional information on how to do this, see the Windows manual or your UNIX documentation.

Working With Symbols

While you are in any field in the SourceGate II BMD from which symbols can be entered, pressing the ***right mouse button (middle mouse button for UNIX systems)*** opens the Symbol Selection window which allows you to select a symbol for that field. An example Symbol Selection window is shown below:



From this window, you can select one of the pre-defined symbols by clicking on it and then pressing the **Select** button. The selected symbol will then appear in the field from which this window was invoked.

Using Help

Using the Microsoft Windows Help Program, the SourceGate II BMD provides online Help for every window. You can resize, move, tile, or cascade the Help window and the current active window so that you can keep both of them displayed.

There are several ways to obtain online Help:

- Context sensitive Help
- Help for the current active window
- Help menu

Currently, 'context sensitive help' and 'help for the current active window' perform the same function. Help is always provided for the current active window if either of these options is chosen. You can always use the accelerator key F1 to obtain help on the current active window.

Context Sensitive Help

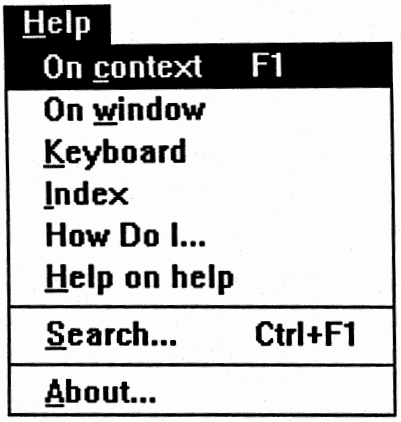
Choose 'Help/On Context'. The SourceGate II BMD will then provide Help for the active window (see ***Active Window Help***).

Active Window Help

Choose 'Help/On Window'. The SourceGate II BMD will then provide Help for the active window.

Help Menu

The Help menu will provide help on all areas. To invoke the Help menu, choose 'Help'. The following menu will then be displayed:



The following is a description of each available menu item:

On Context

Provides help on the current active window (same as ***On Window*** for this release).

On Window

Provides help for the current active window.

Keyboard

Provides keyboard shortcuts (accelerator keys) for the SourceGate II BMD functions and commands. **Note: Accelerator keys are not available when using the UNIX versions of the BMD.**

Index

Displays a list of available help topics from which you can gain access to other parts of the SourceGate II BMD Help System.

How Do I...

Displays a list of common SourceGate II BMD questions and provides answers to each.

Help On Help *(Windows Version Only)*

Switches control from the SourceGate II BMD Help to Windows Help and displays information about using Windows Help.

On Help *(UNIX Versions Only)*

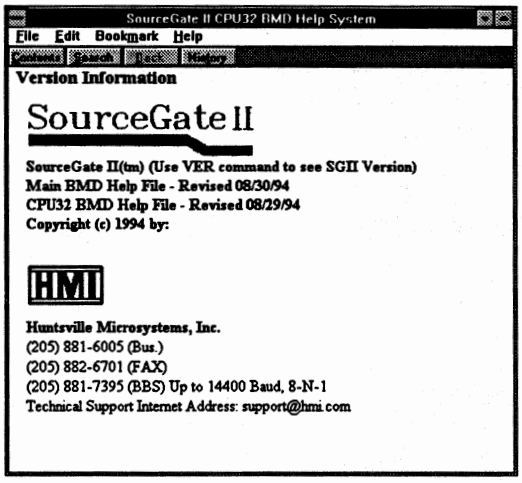
Displays information about how to use the SourceGate II BMD Help System.

Search

Invokes the Help system's Search engine allowing you to search for a particular topic or keyword.

About *(Windows Version Only)*

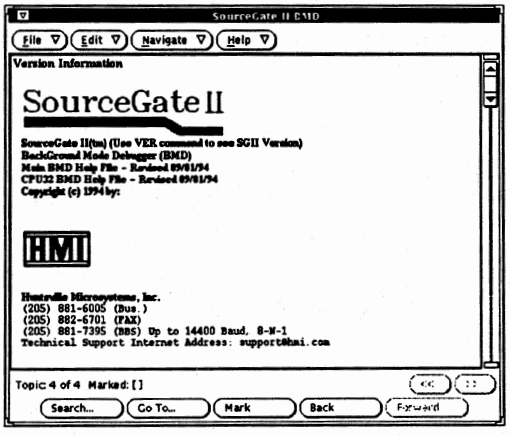
Provides information about the version of the Help files. The following is an example of what you may see when you choose this menu item:


SourceGate II CPU32 BMD Help System
File Edit Bookmark Help
Version Information
SourceGate II
SourceGate II(tm) (Use VER command to see SGII Version)
Main BMD Help File - Revised 08/30/94
CPU32 BMD Help File - Revised 08/29/94
Copyright (c) 1994 by:
HMI
Huntsville Microsystems, Inc.
(205) 881-6005 (Bus.)
(205) 882-6701 (FAX)
(205) 881-7395 (BBS) Up to 14400 Baud, 8-N-1
Technical Support Internet Address: support@hmi.com

Notice that the version of the SourceGate II BMD is not shown in this window. To find out what version of the SourceGate II BMD that you are using, use the "VER" command.

Version *(UNIX Versions Only)*

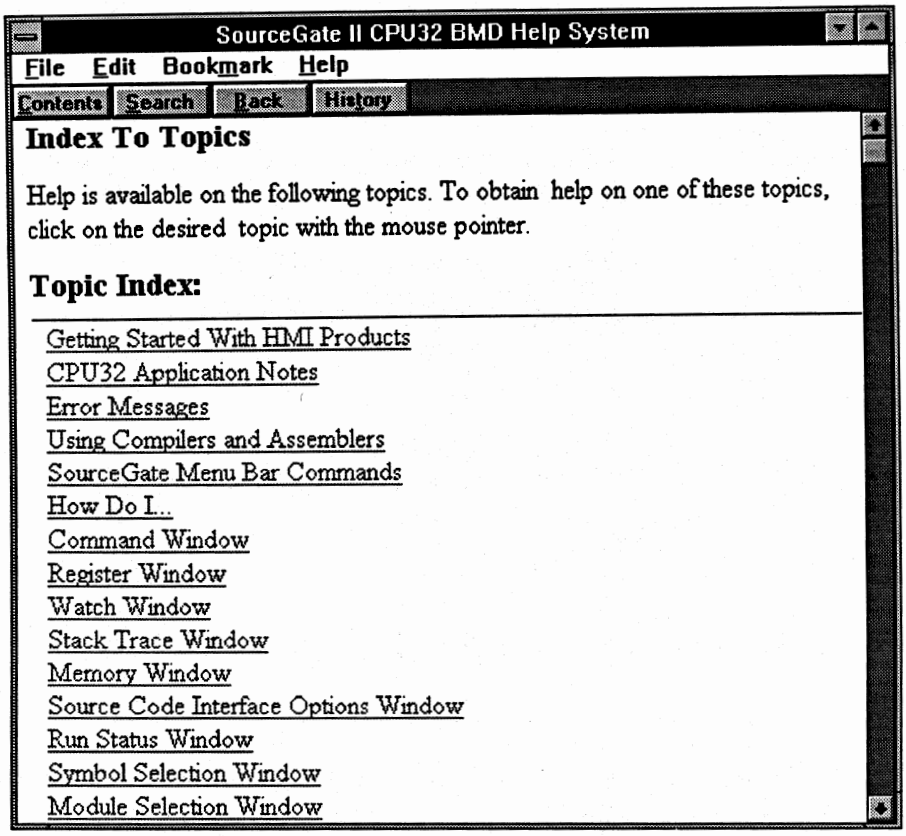
Provides information about the version of the Help files. The following is an example of what you may see when you choose this menu item:


SourceGate II BMD
File Edit Navigate Help
Version Information
SourceGate II
SourceGate II(tm) (Use VER command to see SGII Version)
BackGround Mode Debugger (BMD)
Main BMD Help File - Revised 09/01/94
CPU32 BMD Help File - Revised 09/01/94
Copyright (c) 1994 by:
HMI
Huntsville Microsystems, Inc.
(205) 881-6005 (Bus.)
(205) 882-6701 (FAX)
(205) 881-7395 (BBS) Up to 14400 Baud, 8-N-1
Technical Support Internet Address: support@hmi.com
Topic 4 of 4 Marked: []
Search... Go To... Mark Back Forward

Notice that the version of SourceGate II is not shown in this window. To find out what version of SourceGate II that you are using, use the "VER" command.

SourceGate II BMD Help Window

An example Windows version of the SourceGate II BMD Help window is shown as follows:


SourceGate II CPU32 BMD Help System
File Edit Bookmark Help
Contents Search Back History
Index To Topics
Help is available on the following topics. To obtain help on one of these topics, click on the desired topic with the mouse pointer.
Topic Index:
Getting Started With HMI Products
CPU32 Application Notes
Error Messages
Using Compilers and Assemblers
SourceGate Menu Bar Commands
How Do I...
Command Window
Register Window
Watch Window
Stack Trace Window
Memory Window
Source Code Interface Options Window
Run Status Window
Symbol Selection Window
Module Selection Window

An explanation of the different parts of this window follows:

Help Window Buttons

You can use the Help buttons to display related Help topics. The buttons available differ depending on the platform being used. An explanation of the buttons available for each platform are given in the following sections:

Windows Version

Each button available in the Windows version is described as follows:

Contents

Displays a list of main SourceGate II BMD Help topics from which you can gain access to other parts of the SourceGate II BMD Help System.

Search

Lists all of the Help keywords for the SourceGate II BMD. Enter a keyword or phrase in the 'Search For' text box or select a keyword or phrase from the list box.

Back
Displays the last Help topic you viewed. You can move back through each topic, one at a time. When you are at the first topic you viewed, this button is dimmed.

History
Displays a list of the previous Help topics you viewed. You can choose any topic in the list to display Help for that topic.

UNIX Versions
Each button available in the UNIX versions is described as follows:

Search
Lists all of the Help keywords for SourceGate II. Enter a keyword or phrase in the 'Search For' text box or select a keyword or phrase from the list box.

Goto
Gives you a choice of going to the Index, Glossary, Contents, or Keyboard sections and, if a bookmark has been set, allows you to go to a particular bookmark.

Mark
Marks a help topic as a bookmark. Bookmarks can be recalled at a later date. This provides you an easy way to flag areas of interest and to easily return to these areas at a later date.

Back
Displays the last Help topic you viewed. You can move back through each topic, one at a time. When you are at the first topic you viewed, this button is dimmed.

Forward
If you have previously used the **Back** button, this button allows you to move forward to the next topic that you had previously viewed. If the **Back** button has not been used, this button is dimmed.

Help Window Menus
When the Help window is active, the Help menu bar displays the available menus. The menus available differ depending on the platform being used. An explanation of the menus available for each platform are given in the following sections.

Windows Version
Each menu available in the Windows version is described as follows:

File
Opens Help files, prints Help topics, sets printer options, and closes Help.

Edit
Copies Help text to the Clipboard and adds annotations to Help text.

Bookmark
Places bookmarks in and removes bookmarks from Help topics. You use a bookmark to mark a specific Help reference.

Help
Switches control from the SourceGate II BMD Help to Windows Help and displays information about using Windows Help. To return to the SourceGate II BMD Help, choose **History** and then choose the desired SourceGate II BMD Help topic.

UNIX Version
Each menu available in the UNIX versions is described as follows:

File
Sets printer options, prints Help topics, and exits Help.

Edit
Copies an entire help topic or part of a help topic to the Clipboard as wrapped and shown in the help window or as unwrapped and unformatted data.

Navigate
Searches for a topic or keyword, goes to a particular help section, sets bookmarks, and goes back or forward to topics. Note that the Previous Topic and Next Topic features are not implemented in this version of SourceGate II.

Help
Provides information on how to use the Help system and about the version of the Help system.

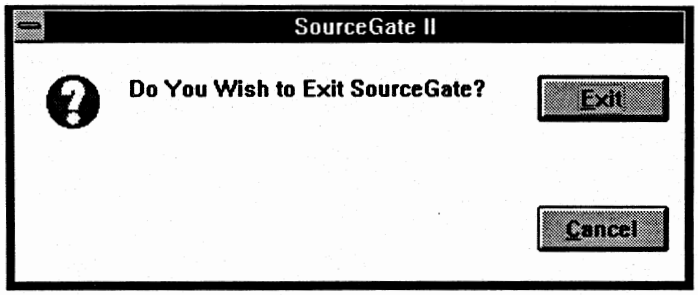
Windows Version Cross-References
Many Windows Help topics contain cross-references to other related Help topics. The cross-references display in green (on a color monitor) with a solid underline. To choose a cross-reference, simply click on it. The SourceGate II BMD will then display the related Help topic.

Exiting The SourceGate II BMD

To exit the SourceGate II BMD, you can do one of the following:

- Choose 'File/Exit SourceGate'
- Press ALT+F4 (Windows version only)
- Double-Click on the SourceGate II BMD window application control box

After performing one of the above steps, you will be asked if you really want to exit the SourceGate II BMD:

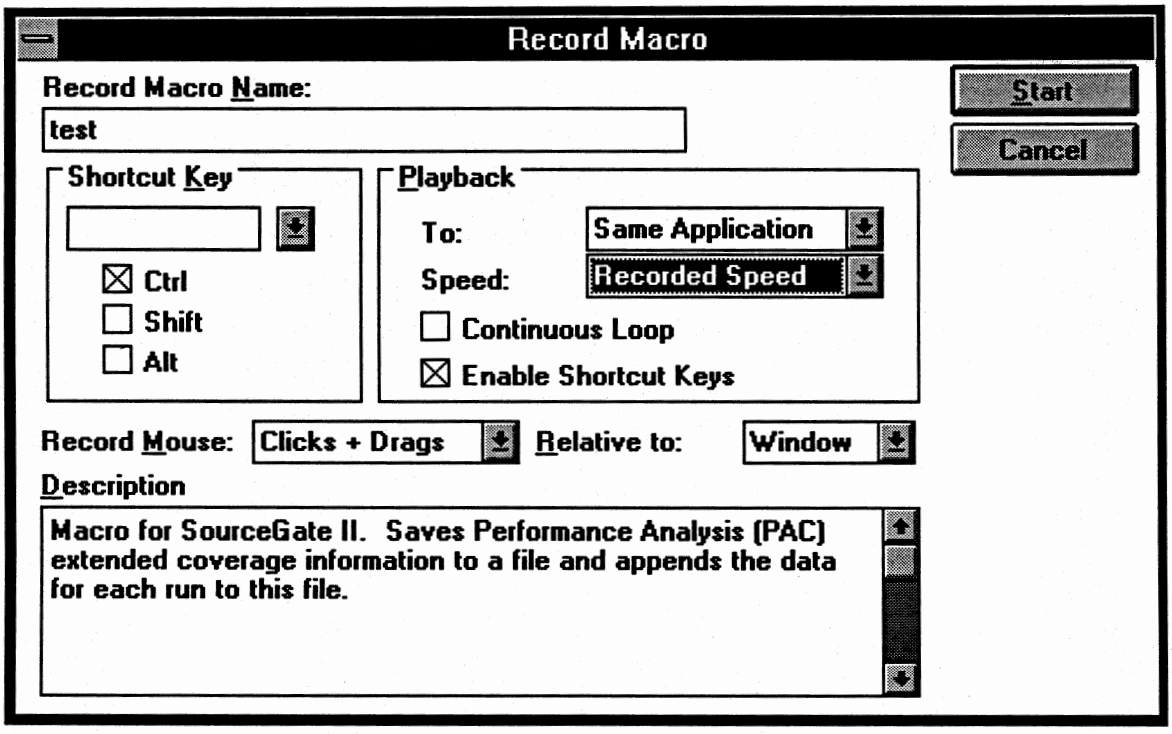


Choosing 'Exit' will exit the SourceGate II BMD. All active window sizes and positions will be saved so that they will appear the same the next time the SourceGate II BMD is started.

Automating The SourceGate II BMD *(Windows Version Only)*

Using the Windows Recorder, the SourceGate II BMD operations can be automated. This will allow you to perform a series of commands, such as opening windows, performing commands in these windows, saving information to a file, etc. This could prove to be invaluable when setting up automatic testing programs.

The following options in the Windows Recorder "Record Macro" window have been shown to work well when recording SourceGate II BMD operations. It is very important that you use the "Recorded Speed" option in order for the playbacks to function correctly. For additional information on using the Windows Recorder, see your Microsoft® Windows manual.



Accelerator Keys *(Windows Version Only)*

You can use accelerator keys, shown beside some commands, to select a command. For example, ALT+F4 will exit the SourceGate II BMD. The following is a listing of all available accelerator key combinations.

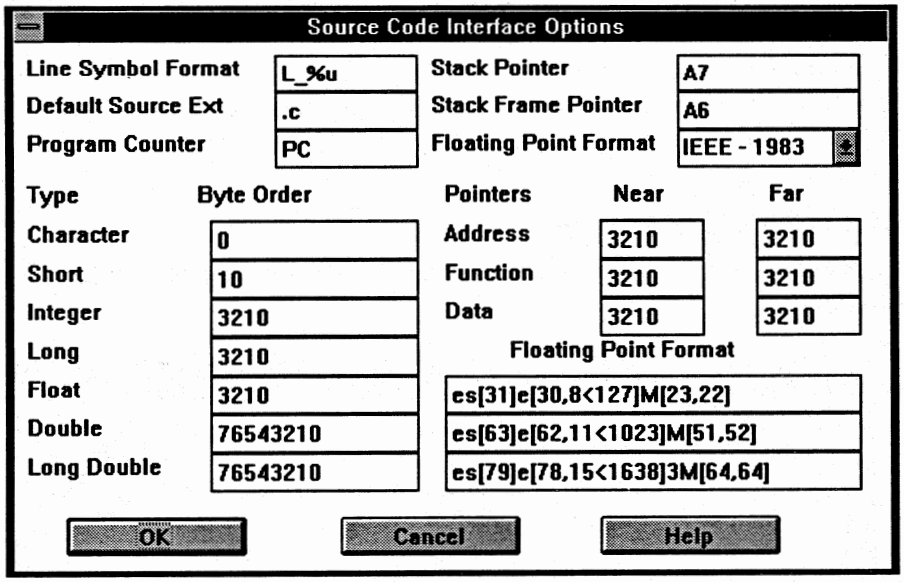
This key...	...does this
Alt+F4	Exit The SourceGate II BMD
Ctrl+A	Activate Assembly Window
Ctrl+C	Activate Command Window
Ctrl+D	Perform 'File/Download Hex File' Function
Ctrl+F	Perform 'File/Execute' Function
Ctrl+H	Peform 'Source/Stop' Function
Ctrl+J	Perform 'File/Record' Function
Ctrl+K	Activate Stack Trace Window
Ctrl+L	Perform 'File/Log' Function
Ctrl+M	Activate Memory Window
Ctrl+N	Activate Tracking Assembly Window
Ctrl+O	Activate Module Selection Window
Ctrl+R	Activate Register Window
Ctrl+S	Activate Run Status Window
Ctrl+U	Perform 'File/Upload Hex File' Function
Ctrl+V	Activate VASM Window
Ctrl+W	Activate Watch Window
Ctrl+Y	Activate Symbol Selection Window
Ctrl+Z	Perform 'Windows/Close All' Function
Ctrl+F1	Perform 'Help/Search' Function
Shift+F6	Activate Previous Window
F6	Activate Next Window
F1	Get Help On Active Window
F5	Perform 'Source/Run' Function From Current PC

3 The BMD Interface

The SourceGate II BMD and the background mode processor that it is controlling have a close relationship. When you enter a command in the SourceGate II BMD, it must be translated into a command that the background mode processor can understand. Likewise, when the background mode processor sends information to the computer, the SourceGate II BMD must take this information and display it to you. This partnership makes up what we refer to as the "BMD Interface". In this chapter, we will attempt to explain the relationship between the background mode processor and the SourceGate II BMD, and we will show you how to set up the SourceGate II BMD to perform the emulation tasks you require. Each area of the "BMD Interface" will be explained in the sections that follow.

Source Code Interface Configuration

The Source Code Interface window is used to configure the SourceGate II BMD to work with almost any compiler. You can define several compiler specific parameters here such as the registers to be used for stack and frame pointers, the line symbol format used by the compiler, etc. To open this window, choose 'Config/Source Code Interface'.



Working With Object Files

The heart of any emulation system is the code that is loaded into memory. In the case of a working target system, this code usually will reside in a programmable device such as a PROM or, in the development stage, may reside in RAM.

SourceGate II BMD can read a variety of object file formats directly into memory without requiring a conversion process. The formats supported include:

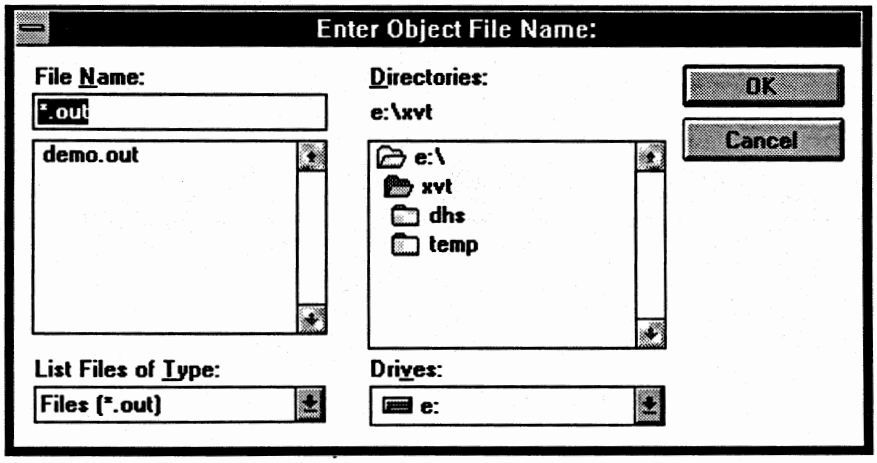
- Motorola S-Record *
- Intel Hex *
- Tekhex *
- HMI Binary *
- Common Object File Format (COFF)
- Software Development Systems (SDS)
- Archimedies Debug Format (IAR)
- IEEE-695
- Extended Tekhex
- Microtec Research Absolute
- Hitachi

  * - No symbol information (including line numbers) supported in this format

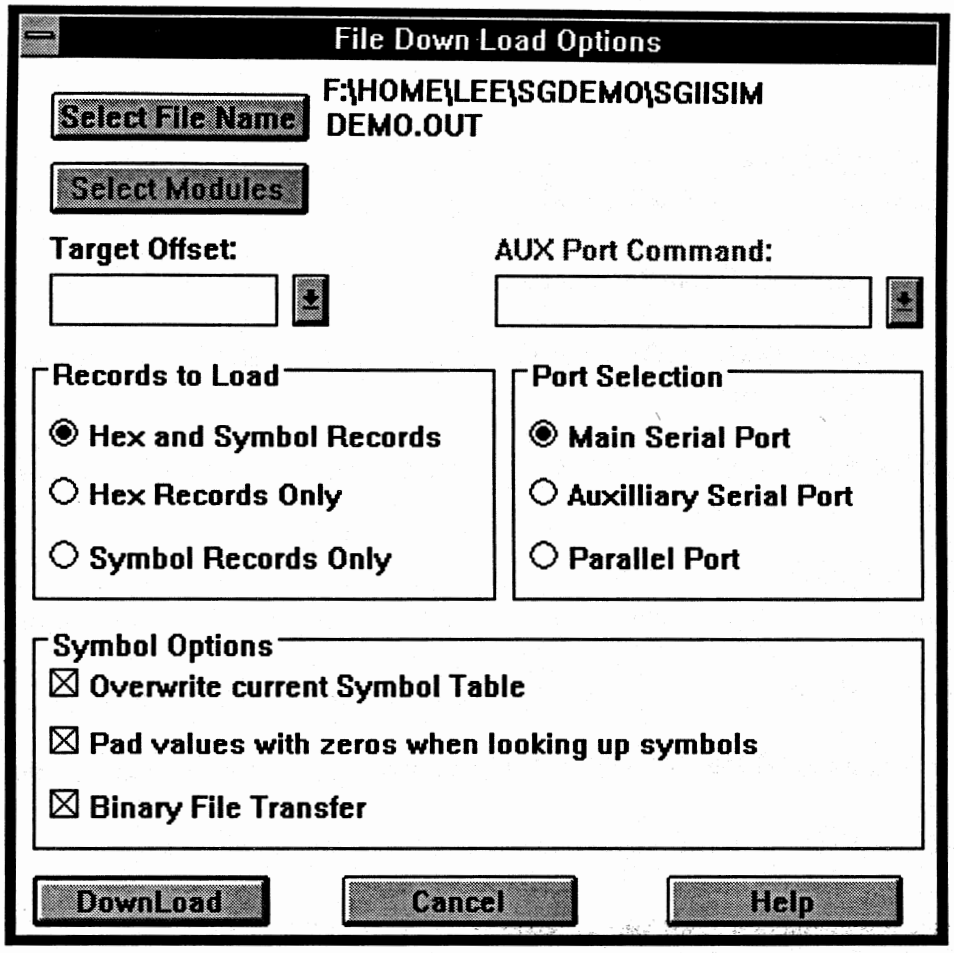
Most compiler/assembler manufacturers provide linkers that can produce one of the above formats. Other formats can be converted to a format that the SourceGate II BMD can read through the use of a Converters Disk supplied free of charge by HMI.

Downloading An Object File

To download an object file into memory, choose 'File/Download Hex File'. You will then be prompted for a filename:



At this point, enter the name of the file to download into memory. Remember that the object file must be in one of the previously listed formats. After you select the file, the "File Down Load Options" window will be shown:



After selecting from the options shown in this window, pressing the **DownLoad** button will download the file into either emulation overlay or target system memory depending on how the memory is mapped in the Configuration window.

File Down Load Options Window Fields

This window allows you to define the file to be downloaded using several options. An overview of each of these options is as follows:

'Target Offset' Field

This field allows you to enter an offset value. This number will shift the load address of each record in the file being loaded. For example, if you enter a target offset value of 8, then a record that is specified as loading at 400H will instead load at 408H.

'AUX Port Command' Field

This field will allow you to enter a command to be sent out the auxiliary port before the read begins. This command may be a string to a mainframe that directs the mainframe to begin sending the file, for example.

NOTE: This field is only changeable if you have "Auxiliary Serial Port" selected in the Port Selection field.

'Records To Load' Fields

These fields allows you to define what records get loaded. The options are as follows:

Hex and Symbol Records - All records in the file will be read.

Hex Records Only - Only the hex code portion of the file will be read. All symbol information is ignored.

Symbol Records Only - Only the symbol portion of the file will be read. All hex code information is ignored.

'Port Selection' Fields

This field allows you to define which port is used for the download. The options are as follows:

Main Serial Port - The file will be downloaded using the main serial port.

Auxiliary Serial Port - The file will be downloaded using the auxiliary serial port.

Parallel Port - The file will be downloaded using the parallel port.

Note: The only valid option that can be used with the BMD is the "Main Serial Port" option. All other options are invalid.

'Symbol Options' Fields

If symbols are being downloaded, these options influence their transfer. The options are as follows:

Overwrite Current Symbol Table - Any current symbol table located in memory will be overwritten with the new symbol table information. Otherwise, the new symbol table will be added to the old information.

Pad Values With Zeros When Looking Up Symbols - All symbols read will have zeros padded at the beginning of their address. For example, a symbol with the value of 400H will be stored as 00000400H in memory. This feature can help qualify addresses in the trace and assembly windows which eliminates erroneous listings of symbols in these windows.

Binary File Transfer - All files will be converted to binary before transmitting. This will speed up the file transfer over transmitting in ASCII.

File Down Load Options Window Buttons

The following buttons are available in this window:

Select File Name

This button allows you to specify the path and file to be downloaded.

Select Modules

This button allows you to specify the individual modules to be loaded from a file. This can help speed up file transfers due to the loading of select module symbol information.

NOTE: This is a future enhancement not supported in this release of SourceGate.

DownLoad

This button begins the file downloading process. This button will be greyed out if no filename has been selected.

Cancel

This button cancels the file loading process.

Help

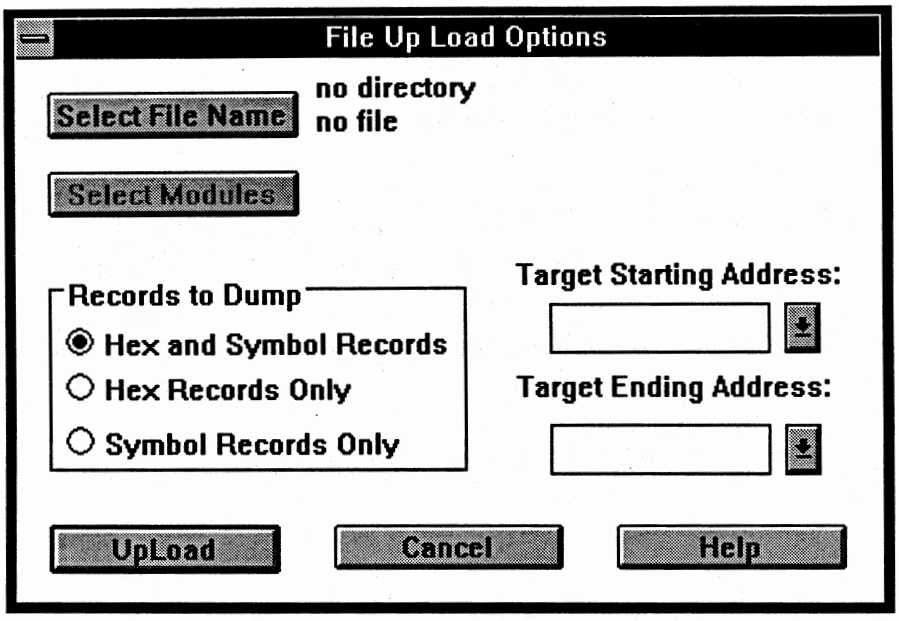
This button invokes help for this window.

Scroll Box

The **Scroll Box** button allows you to choose from previously entered target offset values or AUX port commands.

Uploading An Object File

After you have a working version of your software, you probably will want to save it to a file so that you can burn a PROM and then test it in your system. To do this, choose 'File/Upload Hex File' and enter a filename. You will then be prompted for several parameters as shown below:



File Up Load Options Window Fields

A description of the other fields available in this window follows:

Records To Dump

This section consists of three options which determine the information that is saved as part of the dumped file. The following is an overview of each of these options:

> **Hex and Symbol Records** - Dump both hexadecimal code and any symbol information to the designated file.
>
> **Hex Records Only** - Dump only the hexadecimal code to the designated file. Suppress the dump of any symbol information.
>
> **Symbol Records Only** - Dump only the symbol information to the designated file. Suppress the dump of any hexadecimal code.

Target Starting Address

This field defines the beginning address of the memory to be uploaded.

Target Ending Address

This field defines the ending address of the memory to be uploaded.

File Up Load Options Window Buttons

A description of the functions of the buttons available in this window are as follows:

Select File Name

The **Select File Name** button will prompt you for the name of the file to be saved.

Select Modules

The **Select Modules** button allows you to specify from which modules you want symbol information uploaded into the output file.

NOTE: This is a future enhancement not supported in this release of SourceGate.

UpLoad

The **UpLoad** button begins the upload process to the selected file.

Cancel

The **Cancel** button cancels the upload operation and closes this window.

Help

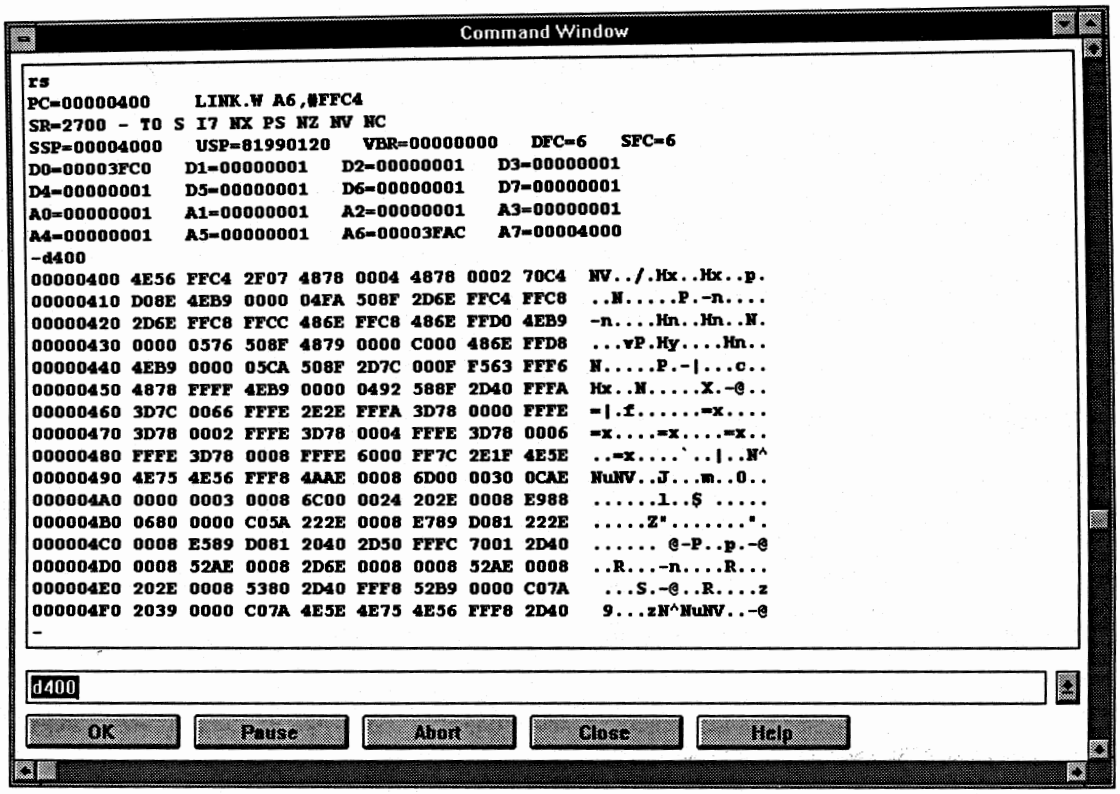
The **Help** button invokes help for this window.

Scroll Box

The **Scroll Box** button allows you to select from previously entered values.

Using The Command Window

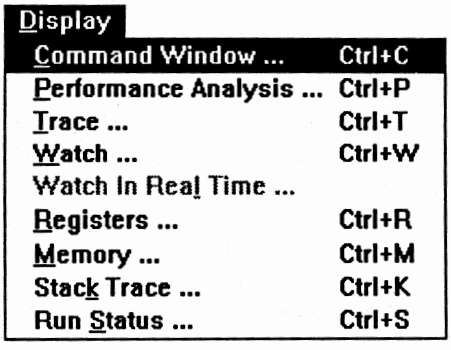
The Command window is the window that you will use to enter commands that affect the background mode processor. BMD responses to these commands will also be shown in this window. The Command window has the following format:



An overview of the operation of the Command window follows. **Note: The Command window examples shown in the following sections were obtained from a 68332 BMD.**

Opening The Command Window

To open the Command window, choose 'Display/Command Window':



An alternate method of opening the Command window is to use the accelerator key combination Ctrl+C (Windows version only).

Entering Command Window Commands

All commands are entered in the command entry field located in the Command Window. Once a command is entered, the ENTER key or OK button will initiate execution of the command. Previously entered commands can be recalled by pressing the **Scroll Box** icon located next to the command entry field. A previously entered command can be highlighted using the mouse or the arrow keys. The highlighted command can be then be executed by pressing the ENTER key or OK button.

In the Command Window, the command abbreviation and the first parameter are not required to be separated, but may be separated by one or more spaces. If multiple parameter fields are used, they must be separated by a recognized delimiter. Allowed delimiters include a comma, a single space, or multiple spaces. If a comma is used as the delimiter, it may be preceded and followed by one or more spaces. This allows you some flexibility in the entering of commands.

Command Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button always executes the command displayed in the command entry field.

Pause

The **Pause** button stops the information from scrolling on the screen. After the **Pause** button is pressed, the button will display **Resume**. Pressing the **Resume** button will allow the scrolling to continue.

Abort

The **Abort** button will abort the command that is currently executing.

Close

The **Close** button will close the Command window.

Help

The **Help** button will invoke Command window help.

Scroll Box

The **Scroll Box** button will allow you to scroll through any previously entered commands.

Using Symbols In The Command Window

Symbols can be entered in lieu of addresses or data values in Command window commands by typing a colon (:) followed by the symbol name. The value associated with the symbol name will be substituted in the command. Also, a number can be added to or subtracted from a symbol value by following the symbol with "+value" or "-value", respectively, where "value" is the number to be added.

The ':' command is used for symbol table display and editing. The 'symbol', '=', and 'value' portions of this command are all optional. This command comes in many different formats. Each of these is discussed in the sections that follow.

:

':' given by itself on a line causes SourceGate to display the entire contents of its internal symbol table.

:symbol

':' followed by a symbol will cause SourceGate to search for and display the first occurrence of the named symbol in the internal symbol table.

:symbol*

':' followed by a symbol with a wildcard character '*' will cause SourceGate to search for and display all occurrences of the symbol name including wildcards. For example, if ":test*" is entered, SourceGate will display all symbols of this type such as "test", "test1", "testit", etc.

:module/symbol

':' followed by a module name, followed by a '/', followed by a symbol name will cause SourceGate to search for the named symbol in or following the named module in the symbol table. If the symbol is found, it will be displayed.

:symbol=value
:module/symbol=value

These ':' commands perform symbol lookup as outlined above, but when the named symbol is found, the old value of the symbol is discarded, and the given value substituted for it.

In the above commands, 'value' defaults to a hexadecimal number unless one of the following characters is used to prefix the constant's value:

%	- Binary override prefix
@	- Octal override prefix
&	- Decimal override prefix
.	- Decimal override prefix
$	- Hexadecimal override prefix
0x	- Hexadecimal override prefix

If the command ":SYM100=0" was entered, the following examples show what the output would be when one of the optional prefixs is used:

:SYM100	SYM100 plus zero offset (output = 0)
:SYM100+10	SYM100 plus 10 hex (output = 10H)
:SYM100+.16	SYM100 plus 16 decimal (output = 10H)
:SYM100+%1000	SYM100 plus 1000 binary (output = 8H)
:SYM100+@10	SYM100 plus 10 octal (output = 8H)

In addition, the symbol can be defined to be the name of another symbol, preceded by a colon. If the named symbol does not exist it will be created. If 'module' is given, the symbol will be created in the named module, otherwise a dummy module name of "USER_DEFINED_MODULE" is assumed.

Using Module Commands In The Command Window

Module commands can be performed while in the Command window. These commands influence the information that is shown in the Command window. The following commands are used to enable a particular level of debugging on a module or globally.

:=A [module] Assembly Mode
:=C [module] Clear Symbol Table
:=D [module] Display Current Modes
:=M [module] Mixed Mode
:=S [module] Source Mode

An explanation of each of these commands is given in the sections that follow.

:=A [module] Assembly Mode Command

The ':=A' command sets the named module (or all modules if none is named) to Assembly Level Debug mode. This is the default mode for all modules when SourceGate is started. While in this mode, only assembly language will be shown in the output of any Command window command, and all source code information will be suppressed.

:=C [module] Clear Symbol Table Command

The ':=C' command clears all symbols from the internal symbol table for the named module (or all modules if none is named). The symbol information for the cleared module must be read back into the internal symbol file in order to perform symbolic/source debugging on that module.

:=D [module] Display Current Modes Command

The ':=D' command displays source debug information for the named module (or all modules if none is named). Source debug information consists of the line number map acquired by SourceGate through examination of the module's line number symbols (whose format is set with the SourceGate command line switch '-L' or through the Source Code Interface Options window), the module's source code file name, and the current debug mode set for that module (Assembly, Mixed, or Source-only).

:=M [module] Mixed Mode Command

The ':=M' command sets the named module (or all modules if none is named) to Mixed Debug mode. Once a module has been enabled for mixed debug mode, that module's source file will be opened, and the source lines for that module will be inserted at appropriate points into the output of all commands which disassemble code (i.e. the 'L' (List) command, 'SS' (single-step) command, etc.).

When the ':=M' command is used, SourceGate must be able to locate and successfully open each named module's source code file. These source code files are assumed to reside in the current directory, and their names are assumed to be the module's name, with the extent of ".C".

If any of these assumptions are false, SourceGate will prompt for a source code file name, giving the name of the module for which it cannot find the source code file. The full path name of the file, along with the corresponding drive designator, should be entered at this time.

If no source code file is available, entering no file name at the prompt will cause SourceGate to set that module back to Assembly mode debug (which requires no source code file), and continue on to the next module to be set to Mixed debug mode.

:=S [module] Source Mode Command

The ':=S' command sets the named module (or all modules if none is named) to Source-only Debug mode. When source-only debug mode is selected for a particular module, all assembly language displays for that module will be suppressed. (Some assembly may be displayed in the source only mode. This assembly is from the library routines of the compiler and is source code to the emulator since most of these routines are written in assembly language). In addition, that module's source file will be opened and the source lines for that module will be inserted into the output of all commands which disassemble code (i.e. the 'L' (list) command, 'SS' (Single-Step) command, etc.).

When the ':=S' command is used, SourceGate must be able to locate and successfully open each named module's source code file. These source code files are assumed to reside in the current directory, and their names are assumed to be the module's name, with the extent of ".C".

If any of these assumptions are false, SourceGate will prompt for a source code file name, giving the name of the module for which it cannot find the source code file. The full path name of the file, along with the corresponding drive designator, should be entered at this time.

If no source code file is available, entering no file name at the prompt will cause SourceGate to set that module back to Assembly mode debug (which requires no source code file), and continue on to the next module to be set to Source debug mode.

Creating Automated Command Files From The Command Window

To begin recording an automated command session file that can be played back later, choose "File/Record". This activates the Command window and prompts you to enter the name of the file in which to store the recorded information. After you have entered the required filename, any commands that are typed in the Command window are recorded in the file until you choose "File/Stop Recording". The file is then closed and can be played back by choosing "File/Execute". **Note: This feature only is supported in the Command window and will not record operations in other windows.**

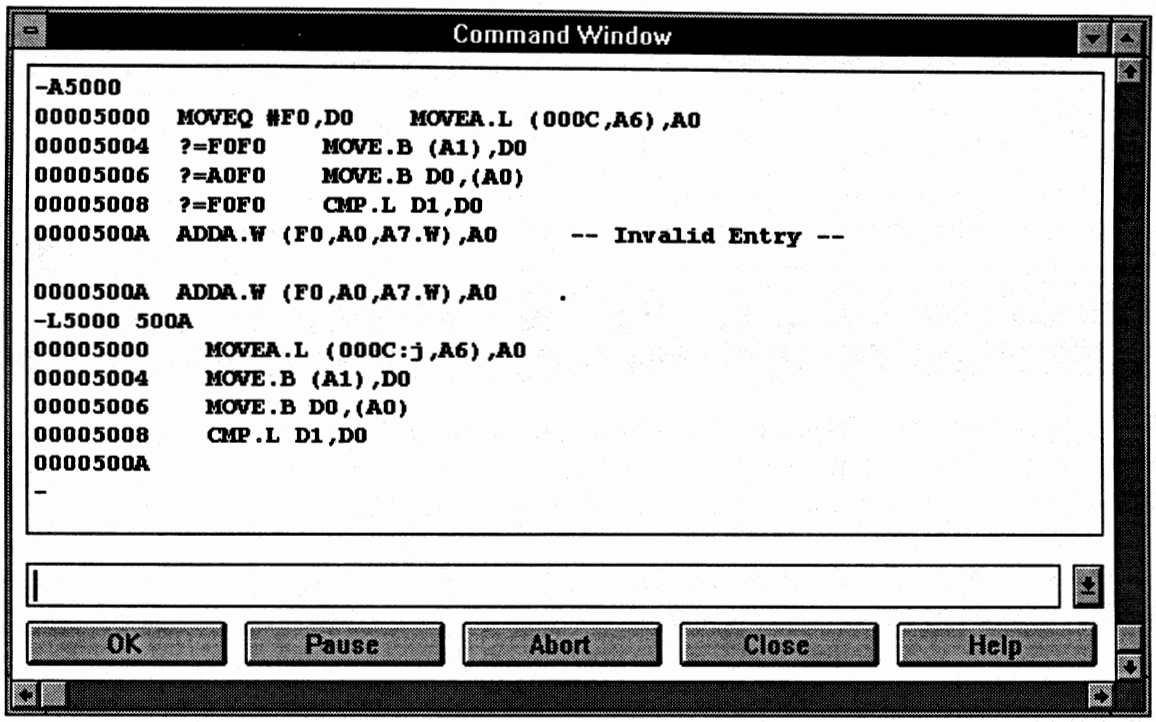
Available Command Window Commands

The following commands are available from the command line for all processor versions of the SourceGate II BMD:

A (In-Line Assembly)
BM (Byte Mode)
C (Compare Memory Block)
CM (Show Status Of Control Board Memory)
D (Dump Memory Block)
DM (Data Memory Mode)
E/EN (Enter/Change Memory)
EXP (Examine/Change Peripheral) Command
EXR (Examine/Change Peripheral Register) Command
F (Fill Memory Block)
G (Begin Real-Time Emulation)
HLD (Hex Listing Disabled)
HLE (Hex Listing Enabled)
I/IN (Input)
L (List Disassembled Code)
LM (Long Word Display Mode)
M (Move Memory Block)
O/ON (Output)
PM (Program Memory Mode)
RD (Read Object File Into Memory)
RP (Reset Processor, Continue Emulation)
RS (Reset Processor, Stop Emulation)
SM (Supervisor Memory Mode)
SP (Stop Emulation)
SR/SRN (Search Block Of Memory)
SS (Single Step)
ST (Show Emulator Status)
UM (User Memory Mode)
VASM (View Assembly)
VER (Display Firmware Version)
W (Write Memory Contents To File)
WM (Word Display Mode)
X (Examine/Change Registers)

All of the preceding commands are described in detail in the sections that follow. Optional parameters are shown enclosed by parenthesis. Any differences between processor versions will be noted in the individual command descriptions.

A (In-Line Assembly) Command



This command allows assembly language mnemonics to be entered into memory starting at the specified address (In-Line Assembly). Once the Assembly command is entered, the address will be displayed with the cursor to the right. At this point, CPU32 assembly language instructions can be entered. After each entry, the BMD displays the next address and prompts for the next instruction. An error in the entered instruction will be flagged by displaying "-- Invalid Entry--" and re-displaying the current address. A period (.) followed by a carriage return terminates the entry sequence and the command.

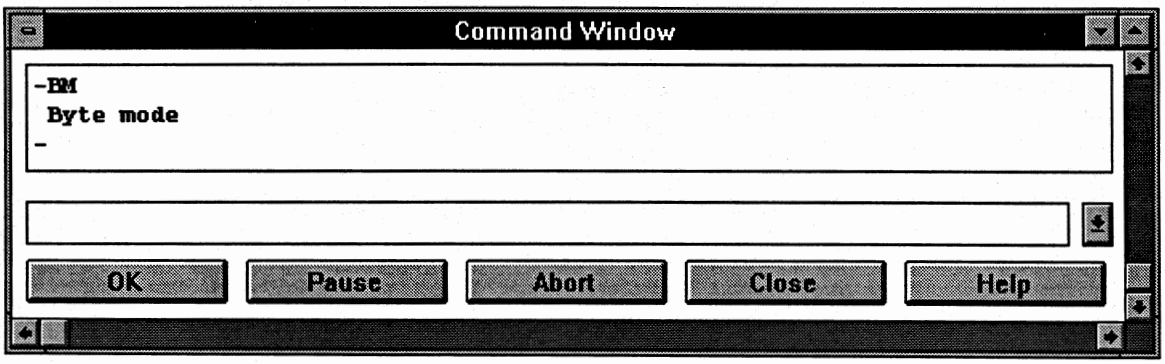
The format for the Assemble command is:

A([SUPD]) Addr

where:

Addr:	is the hexadecimal starting address
SUPD:	are the available overrides

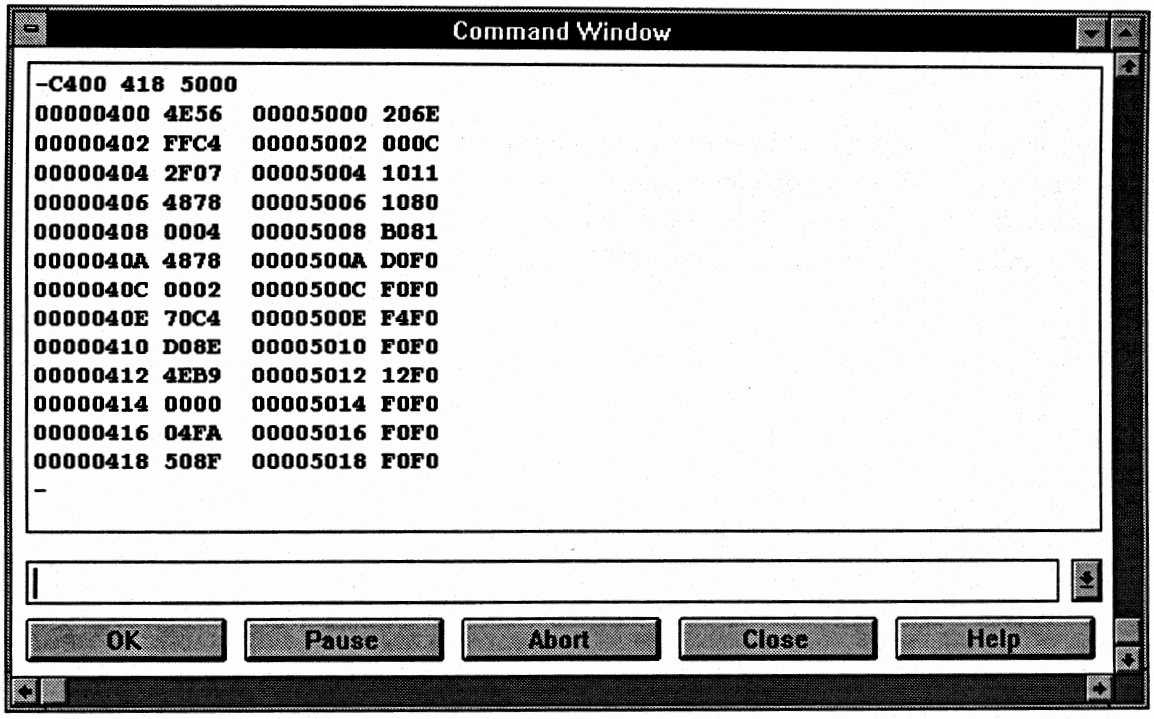
BM (Byte Mode) Command



This command sets the default for input or output values to "Byte Mode". While in Byte mode, commands such as D (Dump memory) will display values in byte mode instead of word mode or long word mode, which are the other choices (see WM and LM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in byte mode, each byte of memory from 8000H through 9000H will contain 33H. If this same command was executed with the BMD in word mode, then 0033H would be stored in each word location from 8000H to 9000H. If this same command was executed with the BMD in long word mode, then 00000033H would be stored in each long word location from 8000H to 9000H.

C (Compare Memory Block) Command



The Compare command will compare one block of memory to another. The first word in the first block will be compared to the first word in the second block, and so on. Any non-comparisons will be displayed on the screen showing the address in the first block with its data value and the address in the second block with its data value. If the two blocks match, then nothing will be displayed. **Note that you can also use the Memory window to compare memory.**

The format for the Compare command is:

C([SUPDR]) Baddr1 Eaddr1 ([SUPD])Baddr2

where:

Baddr1:	is the hexadecimal beginning address of the first compare block
Eaddr1:	is the hexadecimal ending address of the first compare block
Baddr2:	is the hexadecimal beginning address of the second compare block
SUPDR:	are the available overrides

The size of the first block sets the number of words to be compared. For each word in block one that does not match the corresponding word in block two, a line is displayed on the screen.

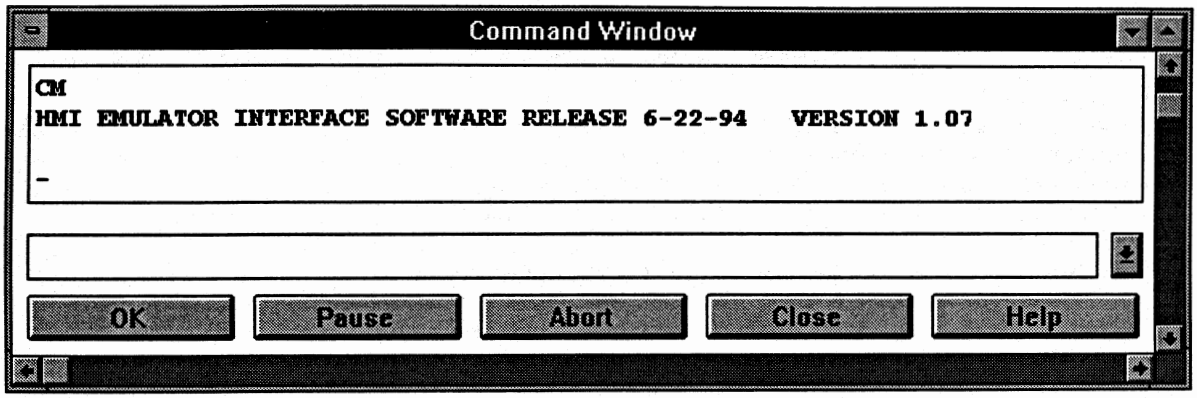
This display line has the format:

Addr1 Value1 Addr2 Value2

where:

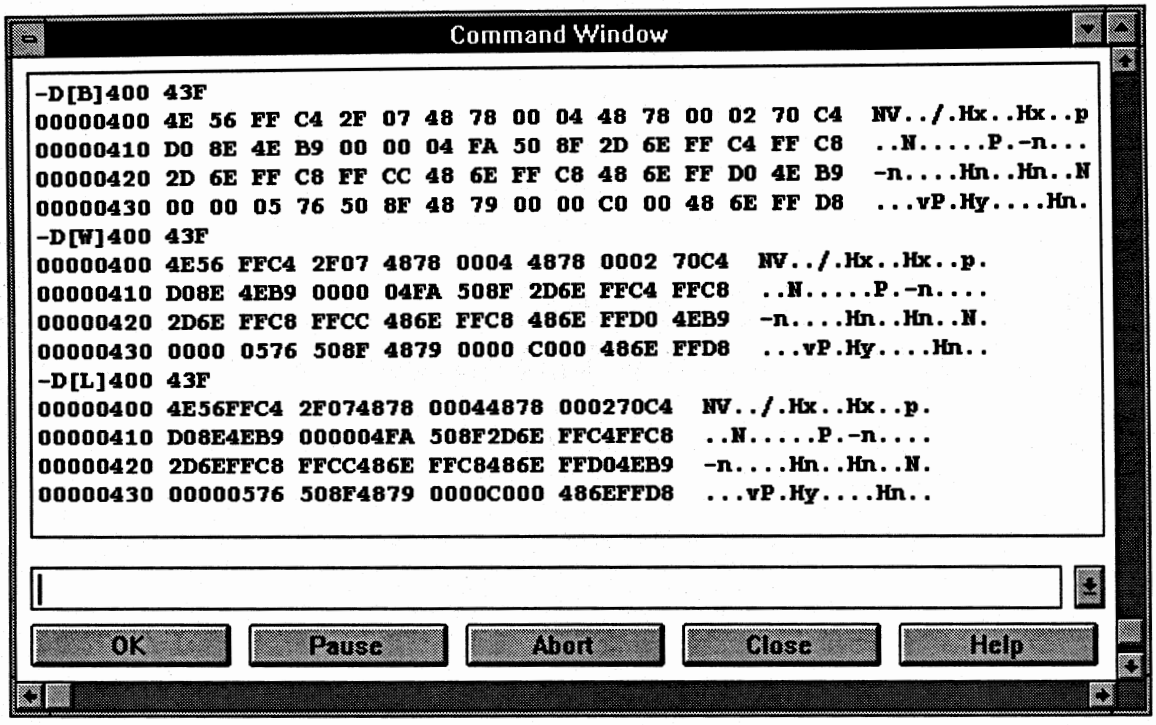
Addr1:	is the address of the word in block one
Value1:	is the contents of addr1
Addr2:	is the address of the word in block two
Value2:	is the contents of addr2

CM (Show Status Of Control Board Memory) Command



Executing this command will show the version of the BMD that is being used.

D (Dump Memory Block) Command



The Dump Command will display the contents of memory in the specified address range showing the hexadecimal data with its ASCII equivalent on the screen. If no ending address is given, then one screen of data will be displayed and the command will terminate. If no starting address is specified then the last ending address (whether user entered or default) will be used as the starting address. Using this method of defaulting allows you to dump a display of memory from one starting address and get the next and subsequent pages of data dumped by simply typing the "D" and a carriage return. **Note that you can also use the Memory window to display the contents of memory.**

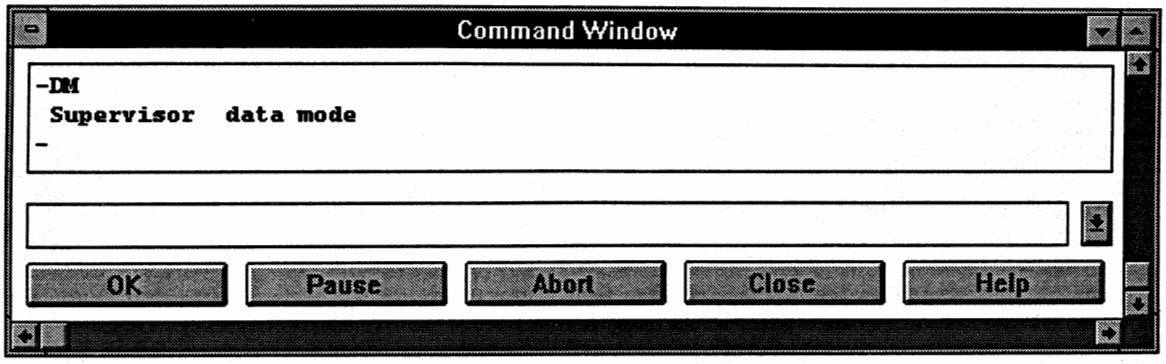
The Format for the Dump Command is:

D([SUPDLWBR]) (Baddr) (Eaddr)

where:

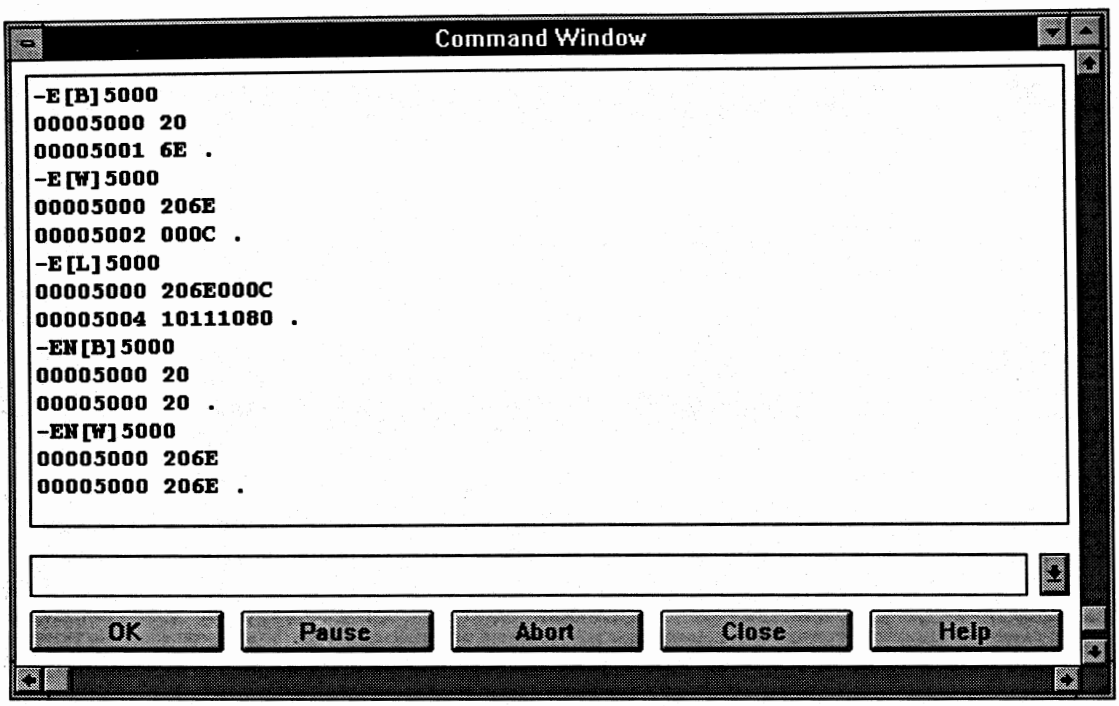
Baddr:	is the hexadecimal beginning address of the block of memory to be displayed
Eaddr:	is the hexadecimal ending address of the block of memory to be displayed
SUPDLWBR:	are the available overrides

DM (Data Memory Mode) Command



The execution of this command sets the default memory category to be Data Memory mode. Using the SM (Supervisor Mode) or UM (User Mode) commands further define the default memory mode.

E/EN (Enter/Change Memory) Command



The Enter Command is used to examine memory at the specified location with the option to change its contents. After executing the Enter Command, the selected address will appear on the screen with the current data. The data can be altered by entering new data and pressing a carriage return, or it can remain unchanged by just typing a carriage return. In either case the next address will be displayed with its data. This command can be terminated by typing a period (.) in the data field followed by a carriage return. **Note that you can also use the Memory window to examine and change the contents of memory.**

There is a variation of the Enter command (EN) that does not advance to the next address. Whether you enter a change and a carriage return or just a carriage return, it will display the value at the specified address only. This feature is very useful in monitoring a memory mapped I/O location.

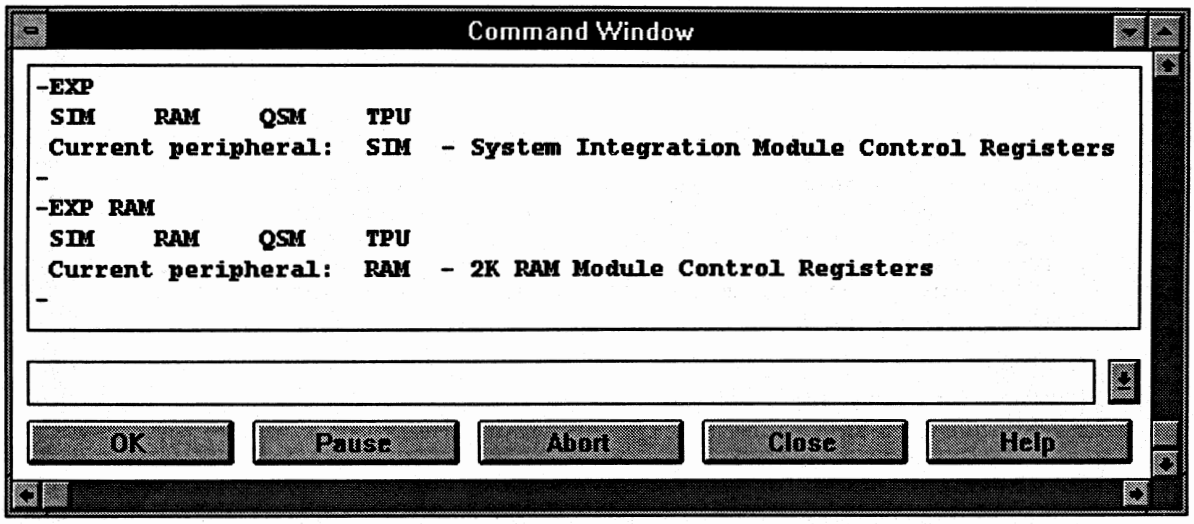
The Format for the Enter Command is:

E([SUPDLWB]) Addr
or
EN([SUPDLWB]) Addr

where:

Addr:	is the hexadecimal address of the data value to be displayed for possible substitution
SUPDLWB:	are the available overrides

EXP (Examine/Change Peripheral) Command



The EXP command changes the current default peripheral. The EXR command can then be used to examine/change a specific register within the default peripheral. Entering EXP by itself on the command line will show the current default peripheral.

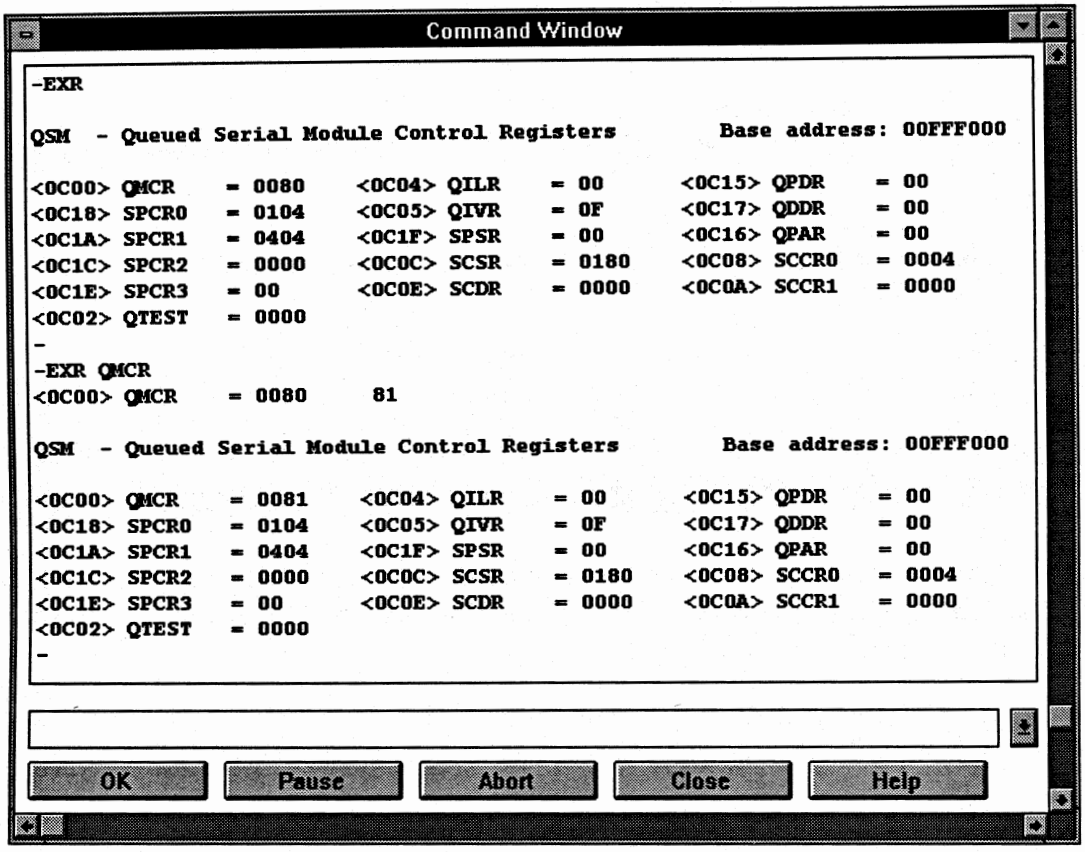
The format of the EXP command is:

EXP per

Where:

per is the desired default peripheral (QSM, RAM, SIM, or TPU for 68331/68332/68340 processors; SIM, CIC, DMA, QDMM, or SCC for 68349/68360 processors)

EXR (Examine/Change Peripheral Register) Command



The EXR command allows a register in the current default peripheral to be examined/changed. The default peripheral can be selected by using the EXP command. Entering EXR by itself on the command line will show all registers associated with the current default peripheral.

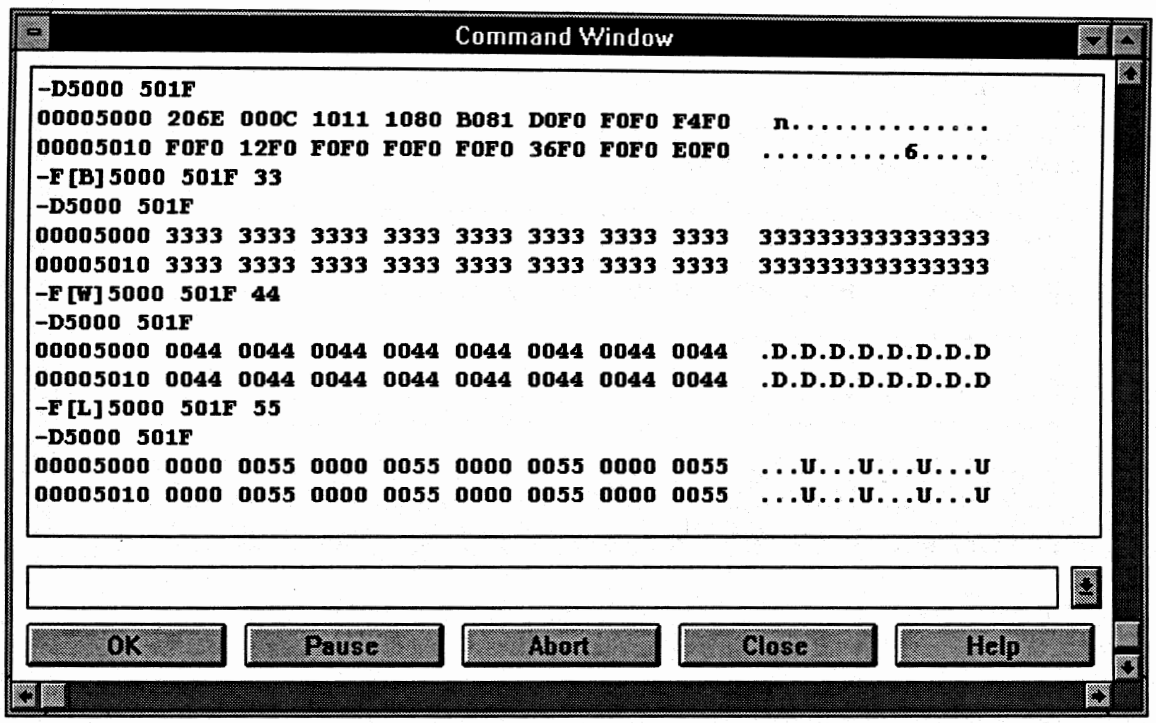
The format of the EXR command is:

EXR reg

Where:

reg is the register peripheral to be examined or changed

F (Fill Memory Block) Command



The Fill Command is used to fill memory with a constant. The given data value will be written to each memory location within the specified range. The value may be a byte or a word in length, determined by the global parameters and overrides used. **Note that you can also use the Memory window to fill memory.**

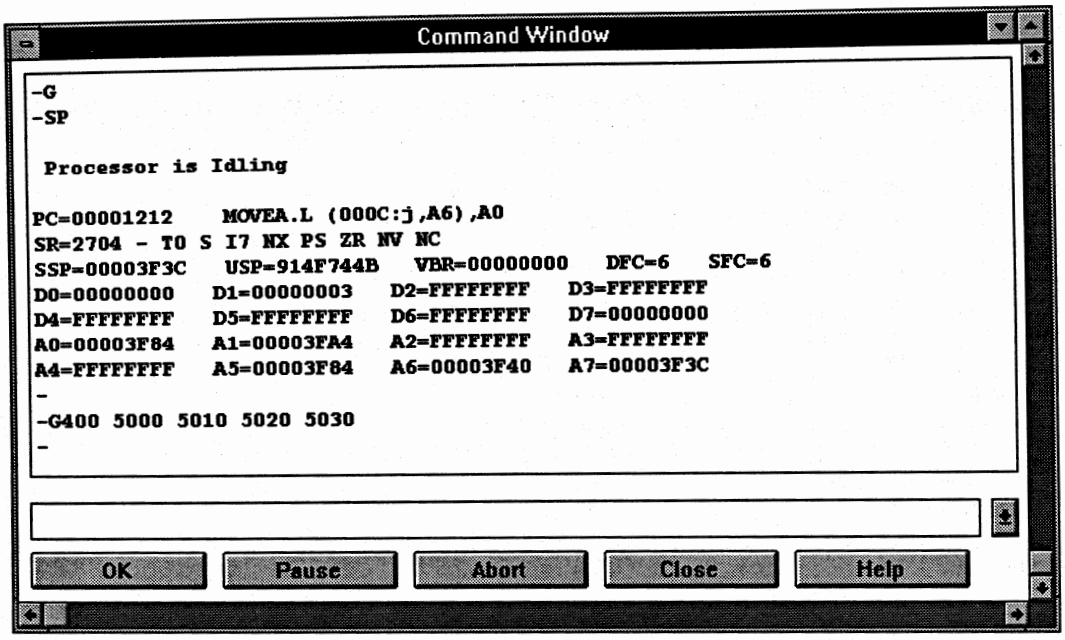
The format for the Fill Command is:

F([SUPDLWBR]) Baddr Eaddr Val

Where:

Baddr:	is the hexadecimal beginning address of the memory block to be filled
Eaddr:	is the hexadecimal ending address of the memory block to be filled
Val:	Constant value to be used to fill the memory block
SUPDLWBR:	are the available overrides

G (Begin Real-Time Emulation) Command



This command is used to start real-time emulation. The options for this command are to run with temporary breakpoints as set in this command, or without a breakpoint. In either case, emulation will begin at the start address given or default to the current program counter. If a starting address is to be used it must be immediately adjacent to the G (otherwise it will be taken as a breakpoint address). **Note: The temporary breakpoint address must reside in RAM since the BMD will attempt to write a BGND instruction there. If the BMD is not able to write to the temporary breakpoint address, the message "Warning: cannot install breakpoint instruction [BGND] at location xxxxxx" will be displayed where "xxxxxx" is the address of the temporary breakpoint.**

Note that you can also use the 'Source/Run' menu item along with the "Run Status" window to begin emulation.

The format for starting emulation with no temporary breakpoints is:

G(Saddr)

The format for setting temporary breakpoints is:

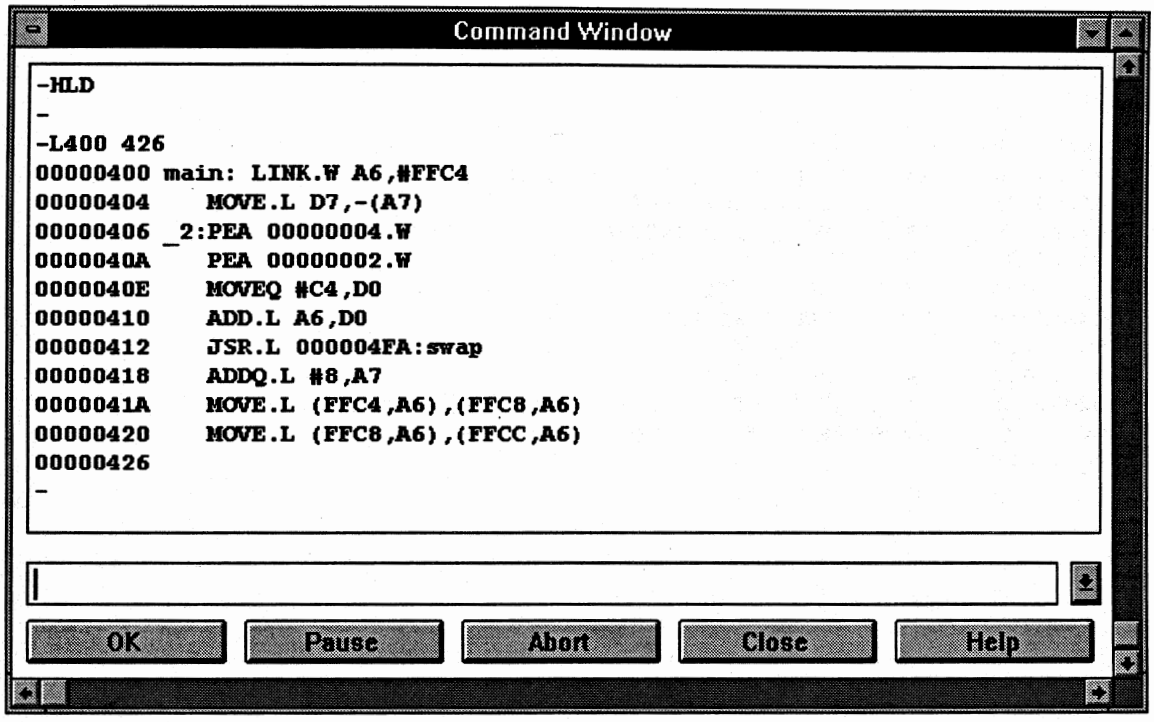
G(Saddr) Braddr1 (Braddr2 Braddr3 Braddr4)

Where:

Saddr: is the hexadecimal address where execution (real-time emulation) will begin

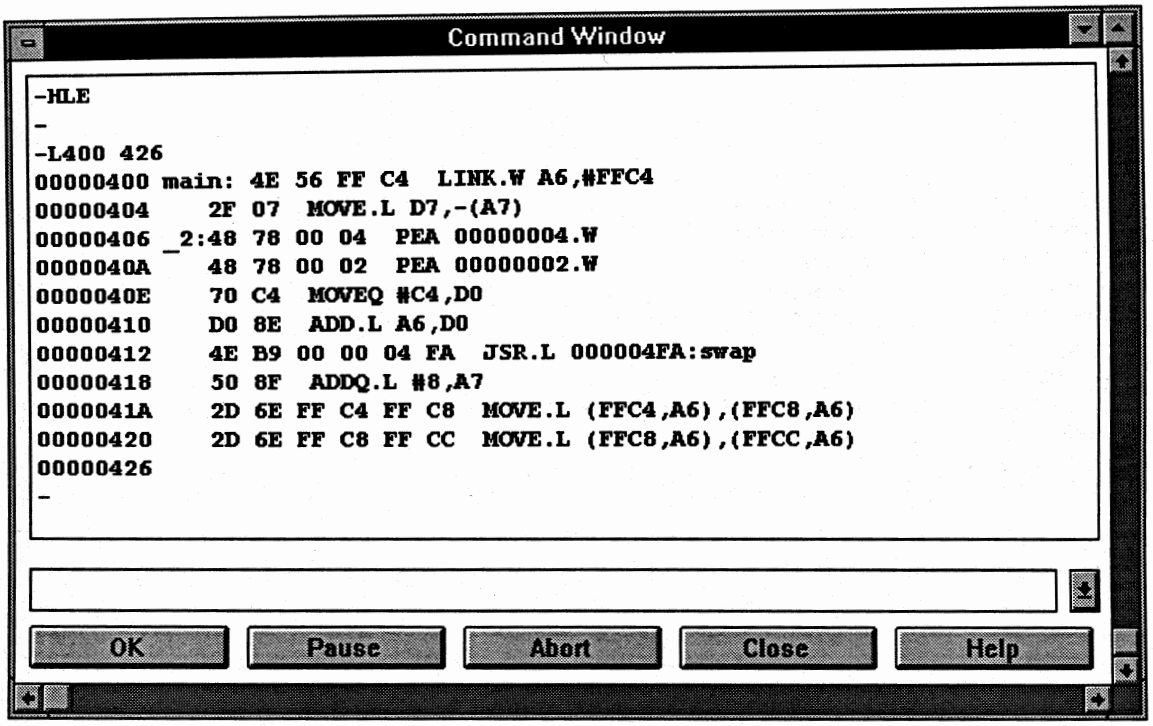
Braddr1-4: are hexadecimal addresses where execution (real-time emulation) will stop (or break) on op-code execution

HLD (Hex Listing Disabled) Command



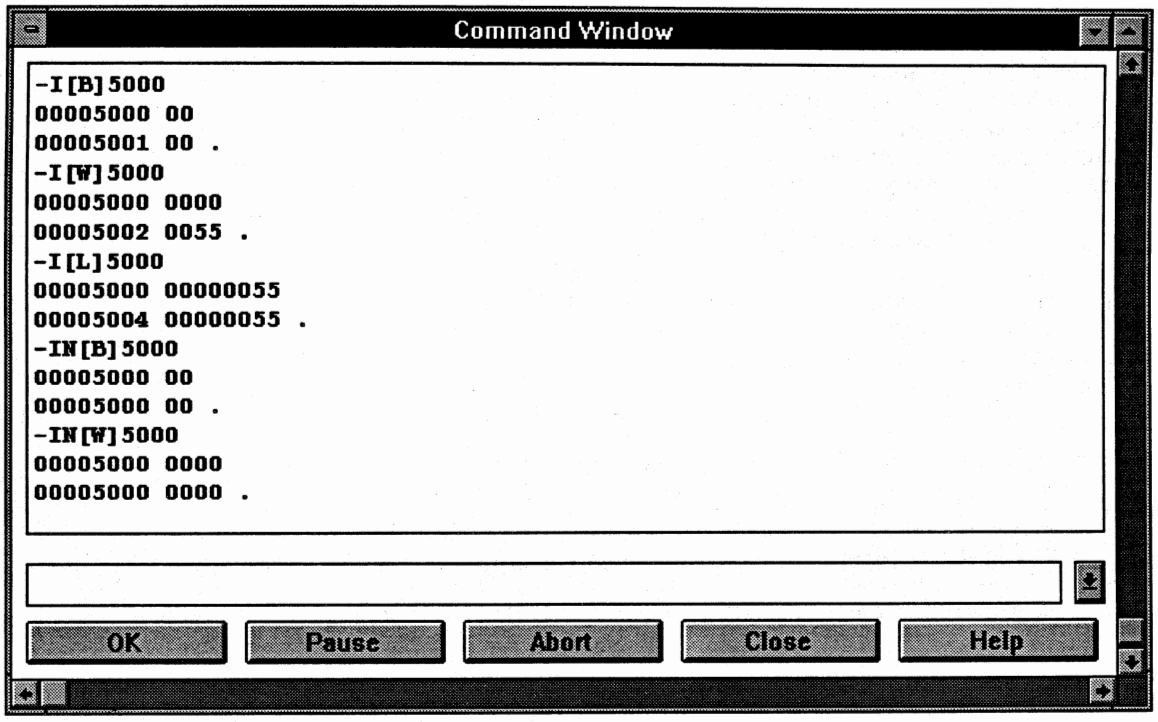
Executing this command will disable the hex information from being displayed with the disassembly information during the L (List) command. To enable the hex information, execute the HLE (Hex Listing Enabled) command.

HLE (Hex Listing Enabled) Command



Executing this command will enable the hex information to be displayed with the disassembly information during the L (List) command. To disable the hex information, execute the HLD (Hex Listing Disabled) command.

I/IN (Input) Command



The I command is used to read and display a certain memory address. This function is very useful when monitoring memory mapped I/O locations. Each time a <cr> is entered, the memory location is read, shown, and incremented to point at the next location. If you wish to suppress the incrementing of the address every time a <cr> is pressed, the IN command should be used. **Note that you can also use the Memory window to read and display memory.**

The format for the Input command is:

I([SUPDLWB]) Addr

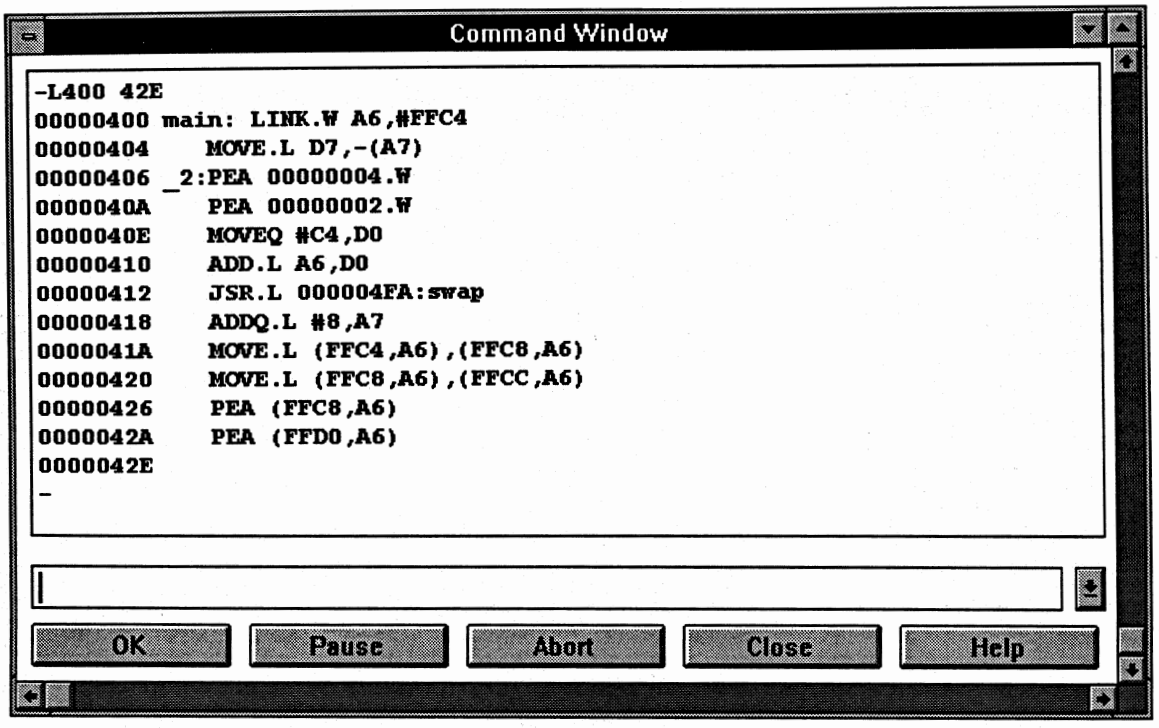
or

IN([SUPDLWB]) Addr

Where:

Addr:	is the hexadecimal address of the data value to be read
SUPDLWB:	are the available overrides

L (List Disassembled Code) Command



This command is used to disassemble the program memory data into assembly language mnemonics. All data and addresses are displayed in hexadecimal. If no ending address is specified, then one screen of disassembled mnemonics will be displayed and the command will terminate. If no beginning address is specified then the last ending address (whether entered or default) will be used as the beginning address. This allows you to disassemble code one display screen at a time by entering the command and an initial starting address for the first display and for the next and each subsequent page only the "L" and a carriage return need be entered. If a range is given, disassembled mnemonics will be displayed from the starting address to the ending address, scrolling the screen if necessary. To pause a scrolling screen enter CNTRL-S and, to resume scrolling a CNTRL-Q. This can also be accomplished by using the "pause" key to stop scrolling and <cr> to resume scrolling. To terminate the command while scrolling depress the space bar or enter a carriage return.

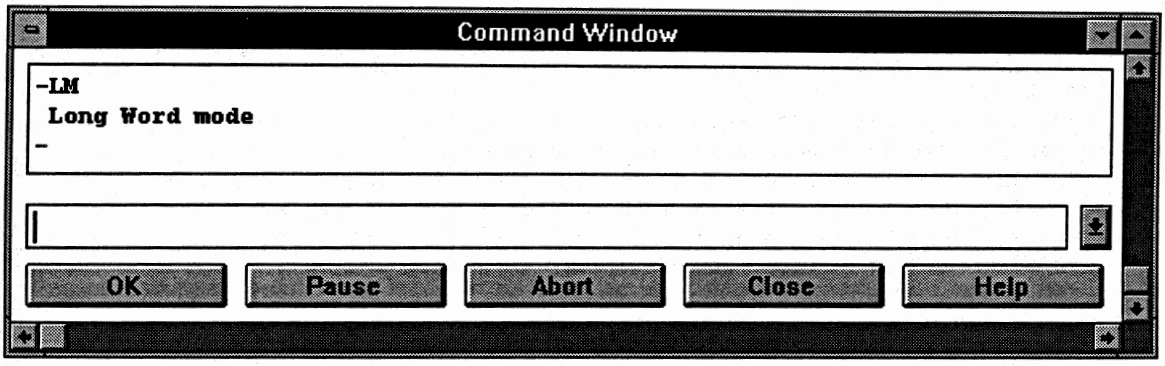
The format for the List Command is:

L([SUPDR]) (Baddr) (Eaddr)

Where:

Baddr:	is the hexadecimal beginning address of the memory data to be disassembled
Eaddr:	is the hexadecimal ending address of the memory to be disassembled
SUPDR:	are the available overrides

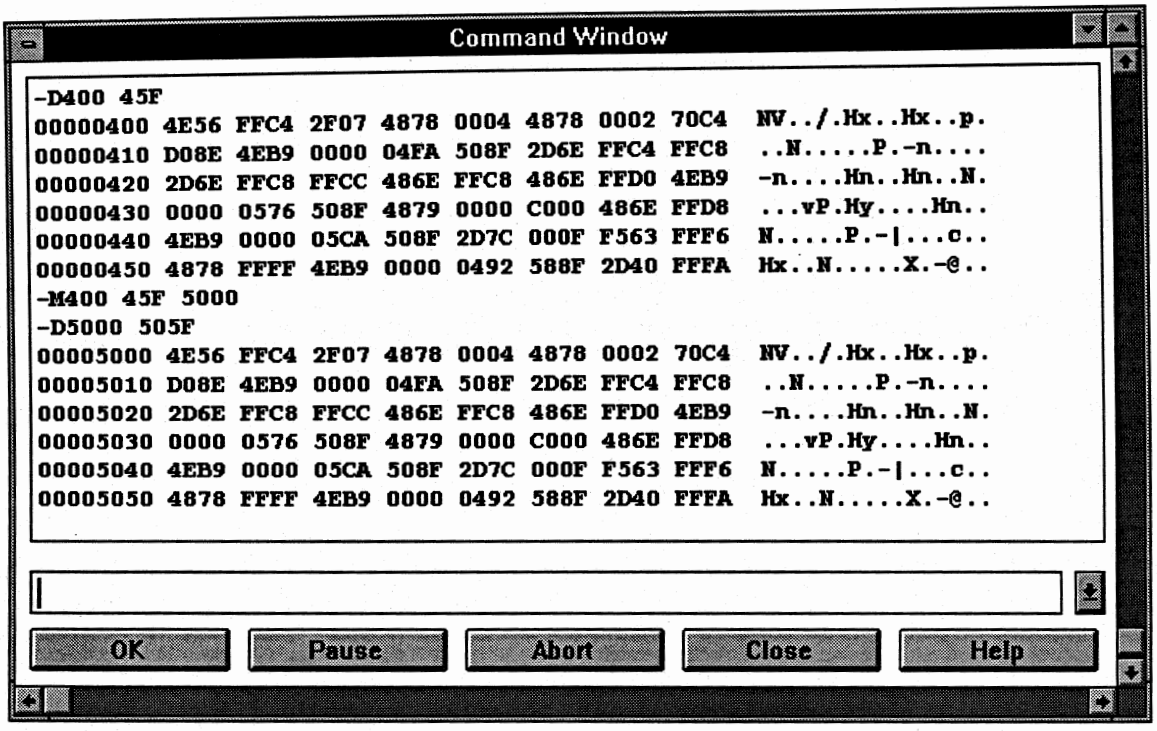
LM (Long Word Display Mode) Command



This command sets the default for input or output values to "Long Word Mode". While in Long Word mode, commands such as D (Dump memory) will display values in Long Word mode instead of word mode or byte mode, which are the other choices (see WM and BM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in Long Word mode, each long word of memory from 8000H through 9000H will contain 00000033H. If this same command was executed with the BMD in word mode, then 0033H would be stored in each word location from 8000H to 9000H. If this same command was executed with the BMD in byte mode, then 33H would be stored in each byte location from 8000H to 9000H.

M (Move Memory Block) Command



The Move Command is used to copy the contents of one block to the contents of another block (non- destructive move). Block one is defined with its starting and ending addresses being the first two address fields displayed, and block two is defined with its starting address being the third address field. The contents of the first memory location of block one is copied to the first memory location of block two, and so on. The next prompt character is not displayed until the move is complete. **Note that you can also use the Memory window to move memory.**

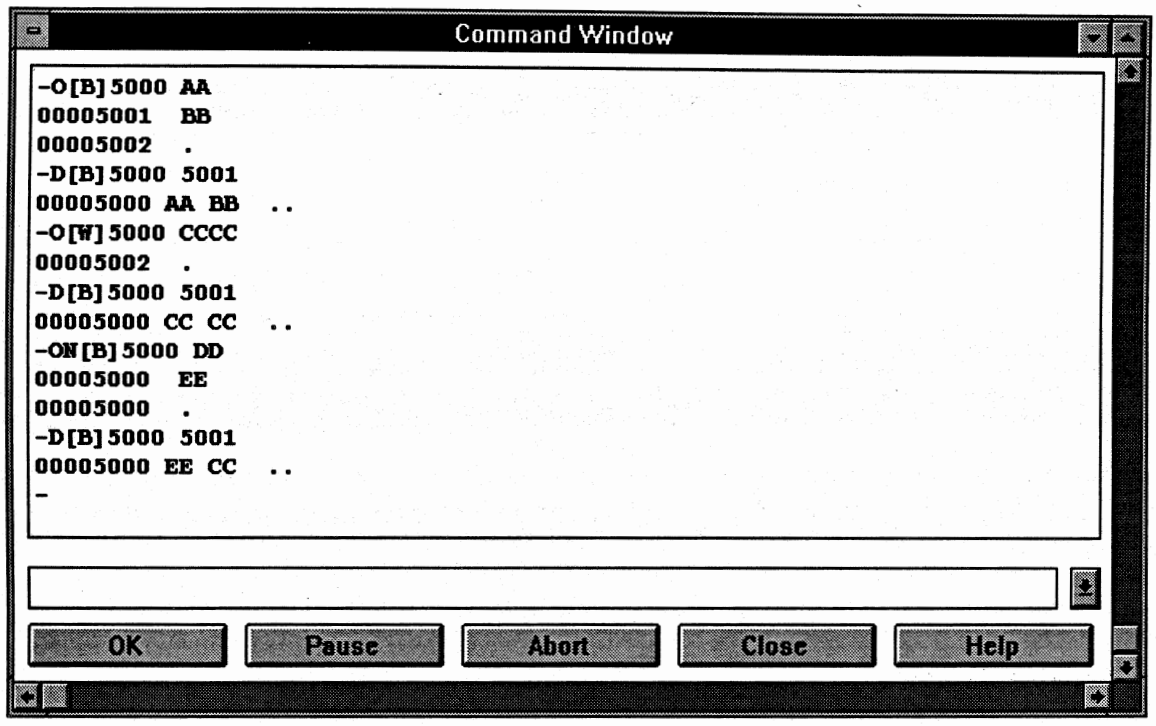
The format of the Move Command is:

M([SUPDR]) Baddr1 Eaddr1 ([SUPD])Baddr2

where:

Baddr1:	is the hexadecimal beginning address of the data "Source" block
Eaddr1:	is the hexadecimal ending address of the data "Source" block
Baddr2:	is the hexadecimal beginning address of the data "Destination" block . The size of the "Source" block defines the number of bytes to be copied into the "Destination" block.
SUPDR:	are the available overrides

O/ON (Output) Command



The O command is used to write to a certain memory address. This function is very useful when testing memory mapped I/O locations. Each time a <cr> is entered, the desired value is written to the displayed address and the address is incremented. The ON command will not increment the address to be written but will write the same address each time a <cr> is entered. **Note that you can also use the Memory window to output the contents of memory.**

The format for the Output command is:

O([SUPDLWB]) Addr,Val

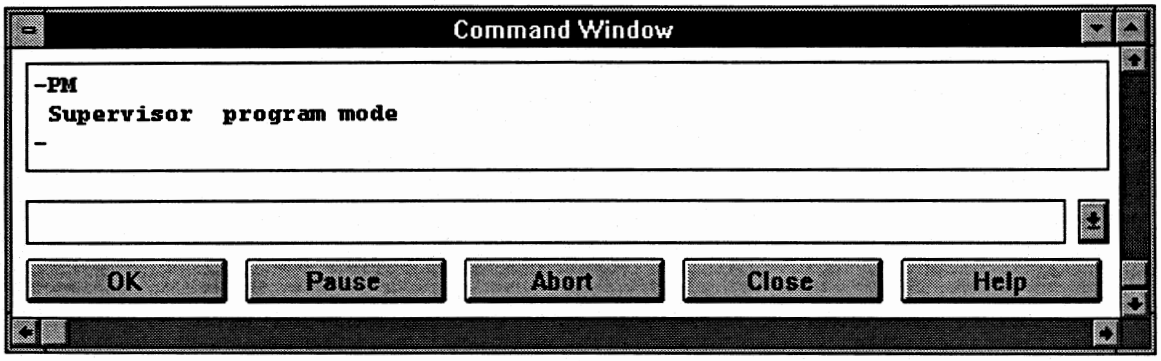
or

ON([SUPDLWB]) Addr,Val

Where:

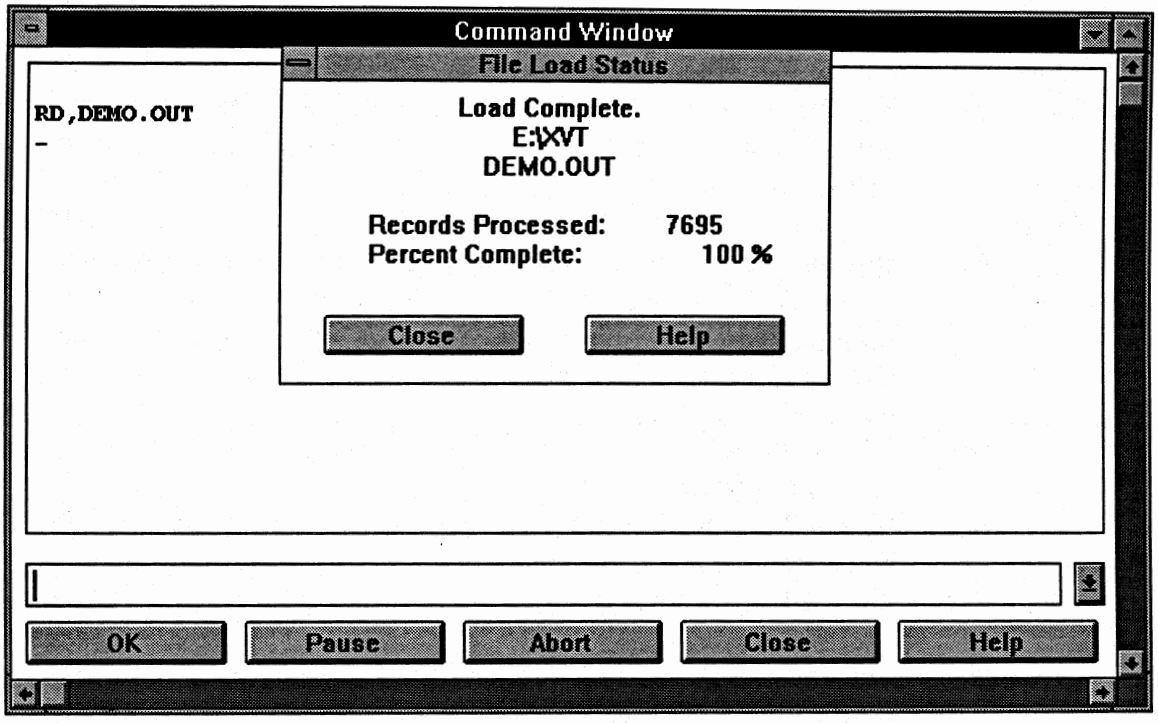
Addr:	is the hexadecimal address to be written with the defined value
Val:	is the hexadecimal value to be written
SUPDLWB:	are the available overrides

PM (Program Memory Mode) Command



The execution of this command sets the default memory category to be Program Memory mode. Using the SM (Supervisor Mode) or UM (User Mode) commands further define the default memory mode.

RD (Read Object File Into Memory) Command



This command is used to transfer a hexadecimal file from a host computer to the target system memory. The file name can be specified with this command. When downloading a file using the SourceGate II BMD, the ASCII file is converted to binary before being transferred to the target system. This essentially doubles the speed of the transfer over the normal ASCII transfer. **Note that you can also use the 'File/Download Hex File' menu item to read a file.**

The file formats accepted by the SourceGate software are Motorola S-Record, Intel Hex, Texhex, HMI Binary, Extended Tek Hex, Microtec Research Absolute, Common Object File Format (COFF), Archimedes Debug (IAR), and Hitachi.

Once the Read Command has been executed, a File Load Status window will appear which will show the status of the read operation. The read operation can be aborted by pressing the **Cancel** button located in the File Load Status window.

When reading a file from the main RS232 port, the format is:

RD([SUPD])(Offset),Filename,(Flags)

Where:

SUPD: are the available overrides

Offset: is a hexadecimal value representing the address offset

Filename: is a file name, valid to the host computer, that designates the file to be downloaded to the target system. If no extension is specified, the default .ABS will be used.

Flags: are options to be used for each read. The options are defined as follows:

- A: Download hex files to the target system using ASCII file transfer
- B: Download hex files to the target system using binary file transfer
- C: Concatenate any symbols found reading a hex file with those which already exist in the internal symbol table (i.e. do not clear the symbol table on a read command)
- D: Use only the digits presented when looking up a symbol by value for display (Opposite of "Z")
- H: Upload/Download only the hex data records present in a hex file. Ignore all symbol records.
- S: Upload/Download only the symbol records of a hex file. Ignore all hex data records.
- M: Toggle between reading "S" record files as Motorola and Hitachi formats. Default is Motorola for Motorola microprocessors.
- Z: Pad a number with leading zeros when doing a lookup by value for symbol display (Opposite of "D").

The default option control flag settings are "BZ".

The 'RD' command opens the named format file (.ABS being the default extension) and reads its symbols into the internal SourceGate symbol table, and its hex data into the target system's memory.

If the optional read bias is given, this offset value will be added to the addresses of all bytes read from the file, before these bytes are stored in memory. This offset is a hexadecimal number. If the filename is not given, the 'RD' command will use the filename given in the last 'RD' command.

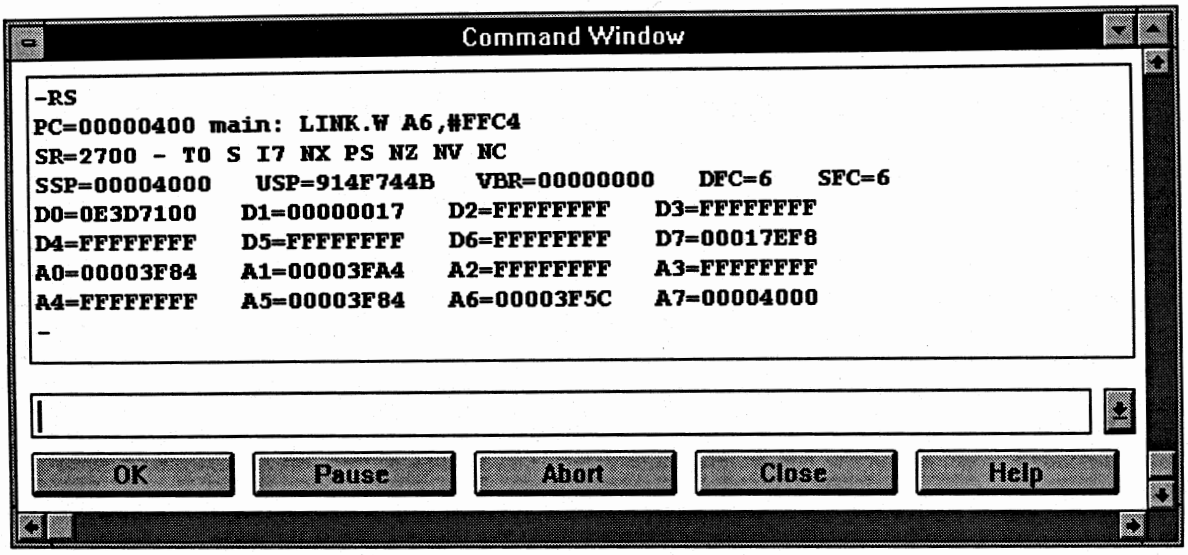
Note: The default file extension used on read (load) and write (dump) operations can be changed by modifying the .SG2 file for your BMD. For read operations, the "DefaultLoadExtension" definition can be changed to reflect the desired extension. For write operations, change "DefaultDumpExtension".

RP (Reset Processor, Continue Emulation) Command

The RP command allows you to reset the processor without stopping emulation.

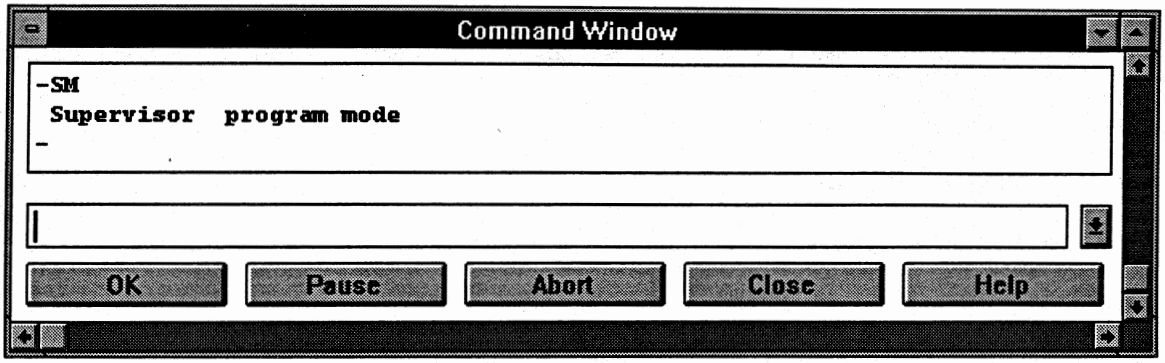
Note: The RP command can only be executed while emulation is in progress.

RS (Reset Processor, Stop Emulation) Command



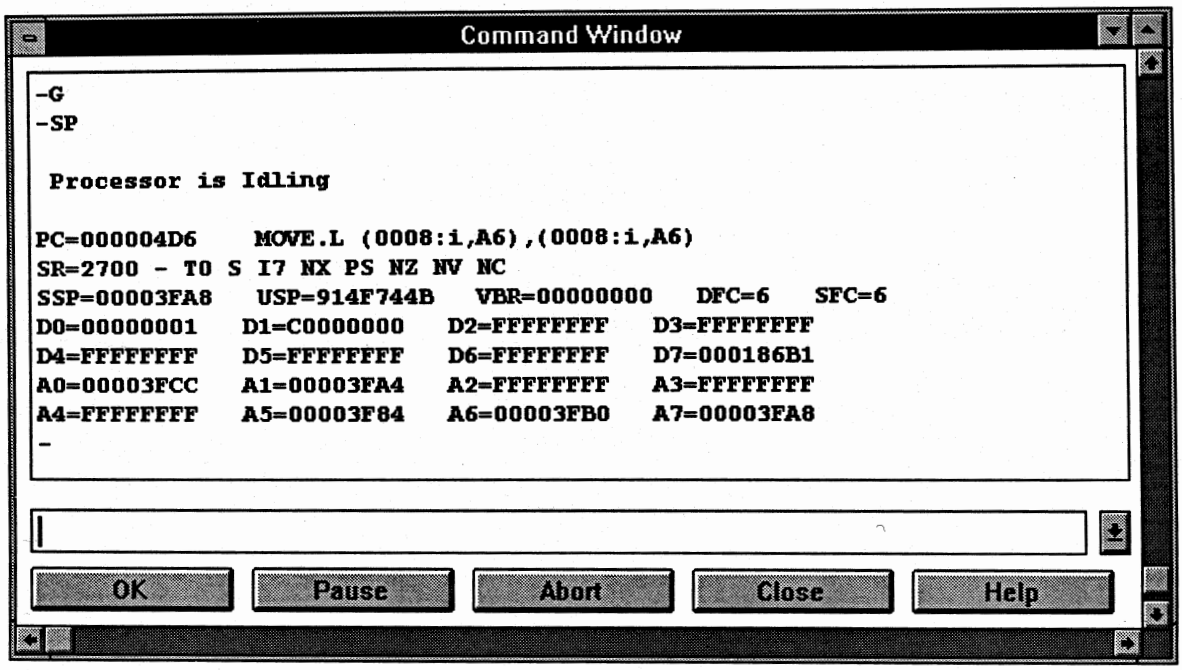
This command will issue a reset to the CPU32 processor. All registers will be updated to the value which they contain upon a processor reset. The Program Counter is initialized to the contents of memory locations 4H and 6H and the Stack Pointer is initialized to the contents of memory locations 0H and 2H. **Note that you can also use the 'Source/Reset Emulator' menu item and the "Run Status" window to reset the processor.**

SM (Supervisor Memory Mode) Command



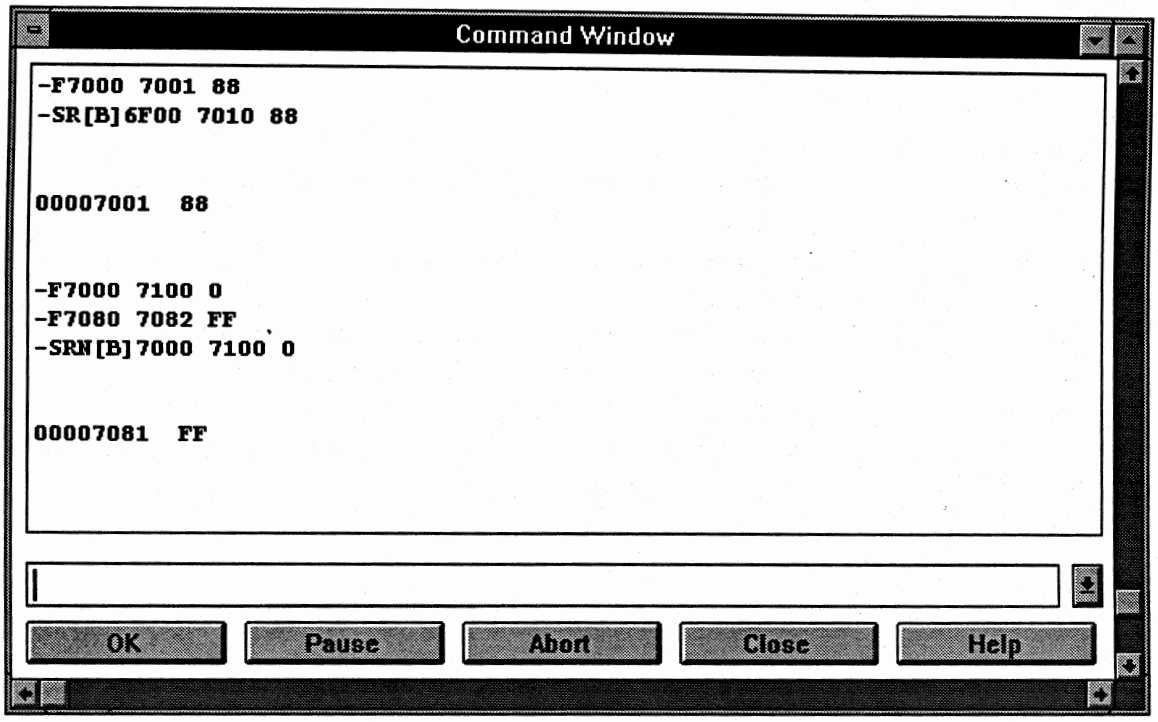
The execution of this command sets the default memory category to be Supervisor Memory mode. Using the PM (Program Mode) or DM (Data Mode) commands further define the default memory mode.

SP (Stop Emulation) Command



This command is used to stop emulation in order to regain control of the processor. Upon entering this command, emulation will break, and the message "PROCESSOR IS IDLING" will be displayed and the register window will be updated with the current register values. **Note that you can also use the 'Source/Stop' menu item and the "Run Status" window to stop emulation.**

SR/SRN (Search Block Of Memory) Command



This command can be used to search memory between two addresses for a particular data pattern. When a match is found, the address is displayed. To search for the next occurrence, hit the Carriage Return key. To terminate this command, type a period. If no match is found, then nothing is displayed. **Note that you can also use the Memory window to search the contents of memory.**

There is also a Search Not option available. In this mode the value is searched for, but the command displays the address of all memory locations in the given range that do not match the search value. This is useful when checking a block of memory that should contain a constant value, such as zero or the value of a previous Fill command.

The format for the Search command is:

SR([SUPDLWBR]) Baddr Eaddr Val

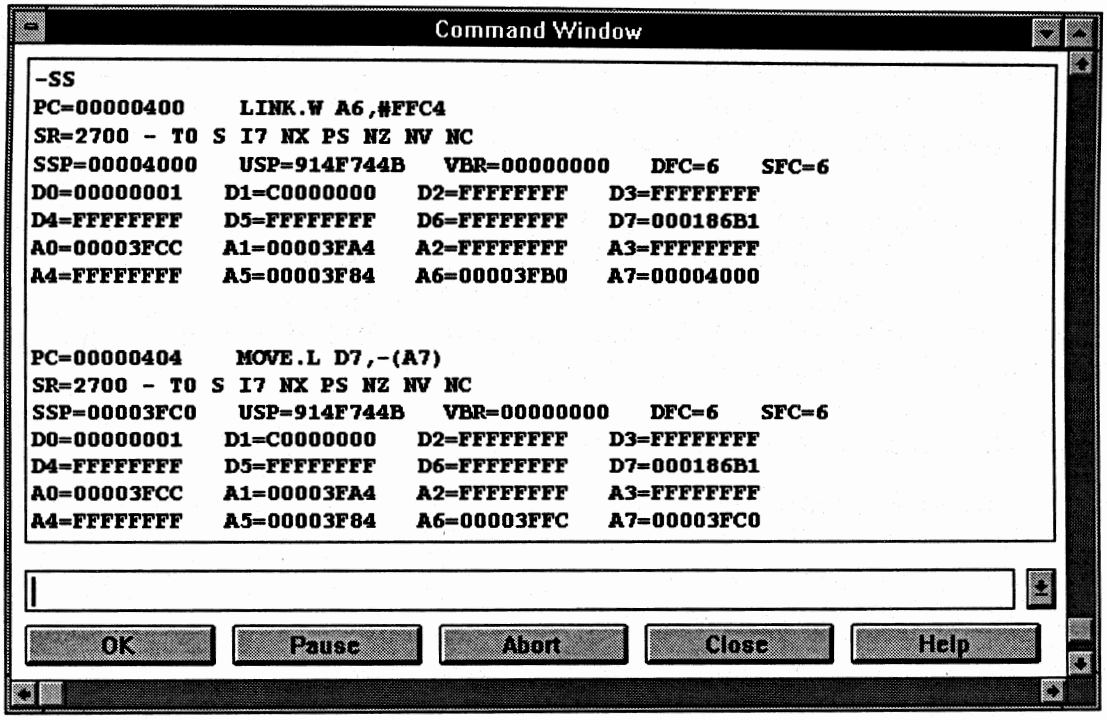
The format for the Search Not command is:

SRN([SUPDLWBR]) Baddr Eaddr Sval

where:

Baddr:	is the hexadecimal beginning address of the block of memory to be searched
Eaddr:	is the hexadecimal ending address of the block of memory to be searched
Val:	is the data value to be searched for and will be 1 byte or word value or a 1-20 character ASCII string enclosed in single quotes.
Sval:	is the data value to be searched for and will be a byte value, a word value or one or two ASCII characters enclosed in quotes.
SUPDLWBR:	are the available overrides

SS (Single Step) Command



This command is used to execute one instruction at a time. Each time a step is performed, the address and instruction to be executed are displayed along with the register values from the previous instruction executed. Therefore, multiple steps will display a history of program flow. There are four options for operating the Single Step Command. The first option is to single step every time a carriage return is entered. To terminate this mode, type a period. The second option is to single step a specified number of times. The third option is to single step continuously. The fourth option allows you to single step "Until" some condition. To terminate the second, third, and fourth options, a carriage return should be entered.

The format of the single step command is as follows:

SS[<count>][X] [Until clause] [Q/Q1/Q2] [Skip]

Where:

SS:	Steps once each time a <cr> is entered. A period (.) aborts stepping
SS<count>:	Steps <count> times without waiting for a <cr>. A <cr> aborts stepping
SSX:	Steps continuously until a <cr> is entered
SS Until clause:	Steps until the condition in the clause is true. A <cr> aborts stepping

The "Quiet" option is invoked by appending a "Q" to the end of the command. In this mode, only the first and last step are displayed (e.g. SSX Q will step automatically until a <cr> is entered showing the first and last instructions stepped). If "Q1" is used, only the last step is shown. If "Q2" is used, all steps are shown in the form "Addr Instruction" where "Addr" is the value of the PC and "Instruction" is the assembly language instruction pointed to by that PC value. In this mode, all register values except the PC are suppressed.

The optional "SKIP" parameter causes all subroutine calls to be shown as a single step. The subroutine called will be executed but the information shown while stepping will not include the instructions in the subroutine, only the call to it. If the subroutine called, in turn, calls other subroutines, or if the subroutine called contains a large number of instructions, there may be a noticeable wait until the program returns to the main program.

The Until clause is defined as follows:

```
OPERAND1 OPERATION OPERAND2

Where OPERAND1 can be:

Reg <reg> - Register name
Long at <addr> - Long word value at address <addr>
Word at <addr> - Word value at address <addr>
Byte at <addr> - Byte value at address <addr>

OPERAND2 can be:

Reg <reg> - Register name
#<value> - Immediate value (size determined by OPERAND 1)
<addr> - Value at <addr> (size determined by OPERAND 1)

OPERATION can be:

 = - OPERAND1 equal to OPERAND2
== - OPERAND1 equal to OPERAND2
!= - OPERAND1 not equal to OPERAND2
<> - OPERAND1 not equal to OPERAND2
>< - OPERAND1 not equal to OPERAND2
 > - OPERAND1 greater than OPERAND2
 < - OPERAND1 less than OPERAND2
>= - OPERAND1 greater than or equal to OPERAND2
=> - OPERAND1 greater than or equal to OPERAND2
<= - OPERAND1 less than or equal to OPERAND2
=< - OPERAND1 less than or equal to OPERAND2
```

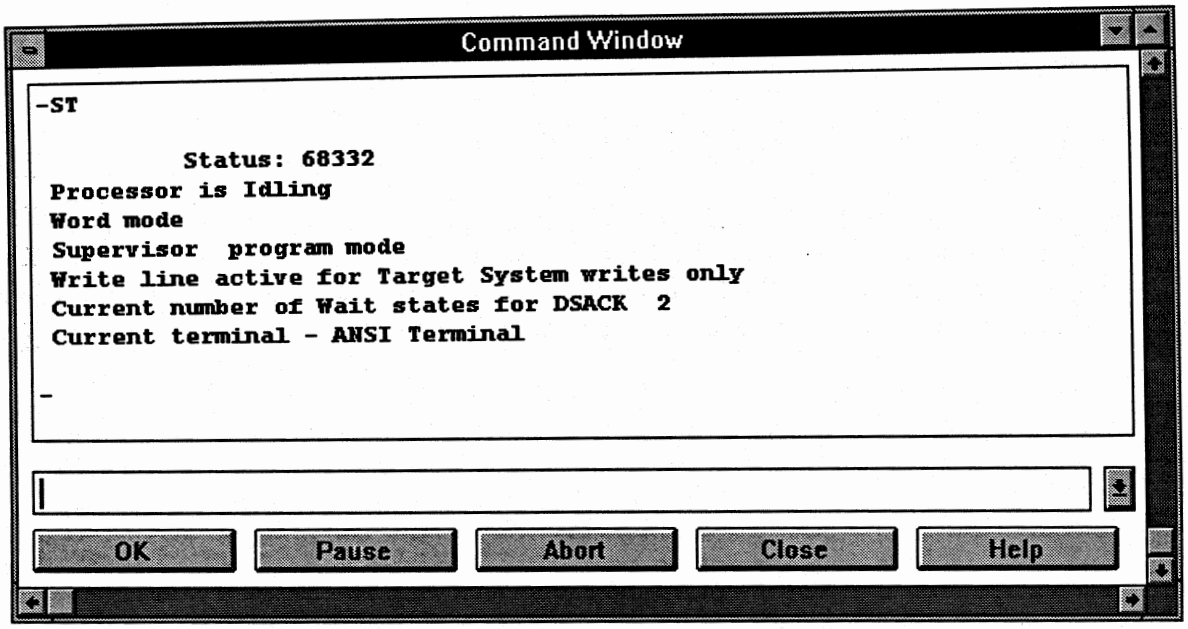
Example:

SS Until word at 5000 = #1234

Would stop stepping when 1234H appears at address 5000H

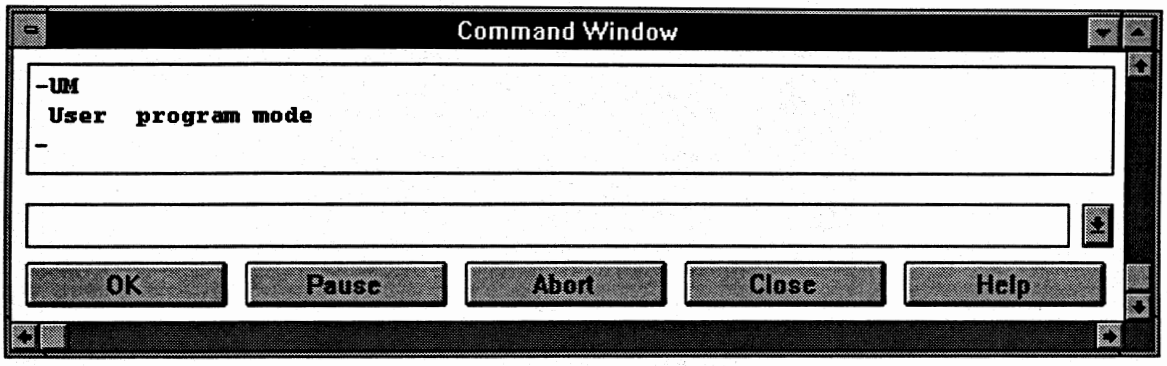
It should be noted that this command takes into account the way data is stored in memory based on the microprocessor manufacturer. If, for example, you wished to stop on a word value of 1234H being in location 5000H, the BMD will look for a 1234H at location 5000H for Motorola processors and 3412H at location 5000H for Intel, Hitachi, and Zilog processors.

ST (Show Emulator Status) Command



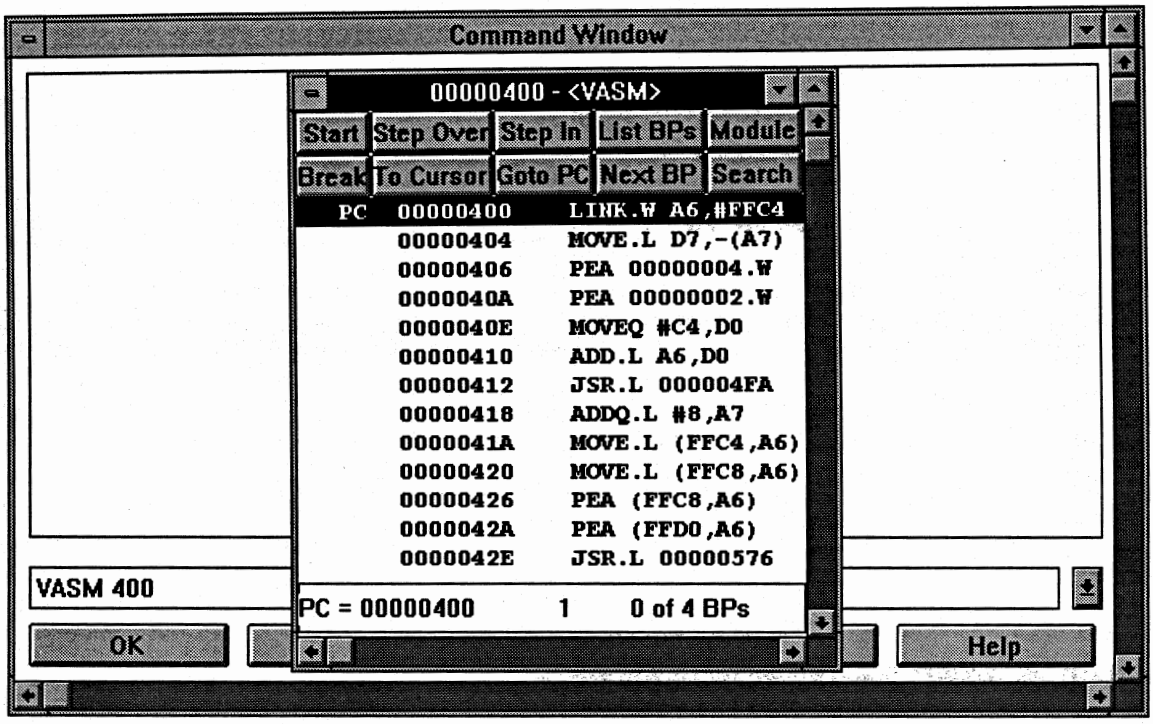
Executing the Status command will display the current status of the BMD.

UM (User Memory Mode) Command



The execution of this command sets the default memory category to be User Memory mode. Using the PM (Program Mode) or DM (Data Mode) commands further define the default memory mode.

VASM (View Assembly) Command



This command will open an assembly Code window beginning at the specified address. **Note that you can also use the 'Source/VASM' menu item to open a VASM window.**

The format of the VASM command is:

VASM <addr>

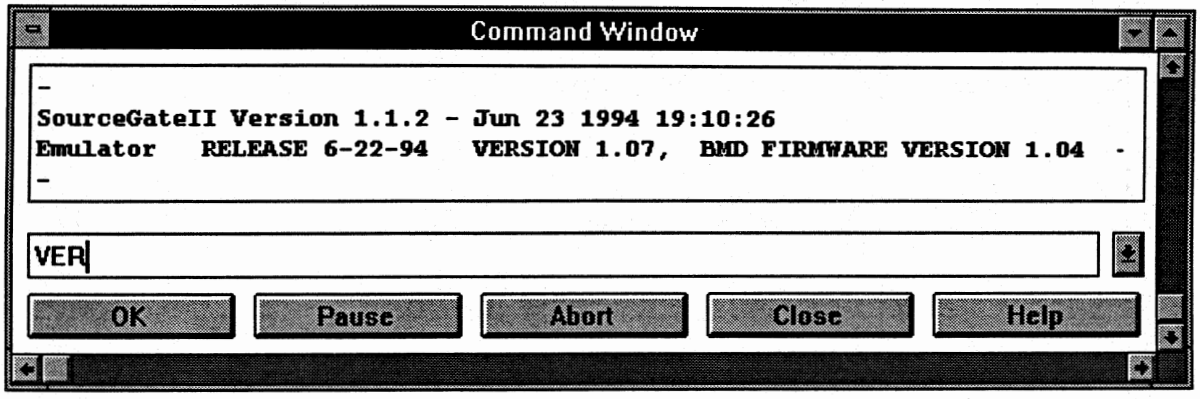
Where:

addr: is the beginning address of the VASM window

If no address is entered, the VASM window will begin with the address of the current program counter.

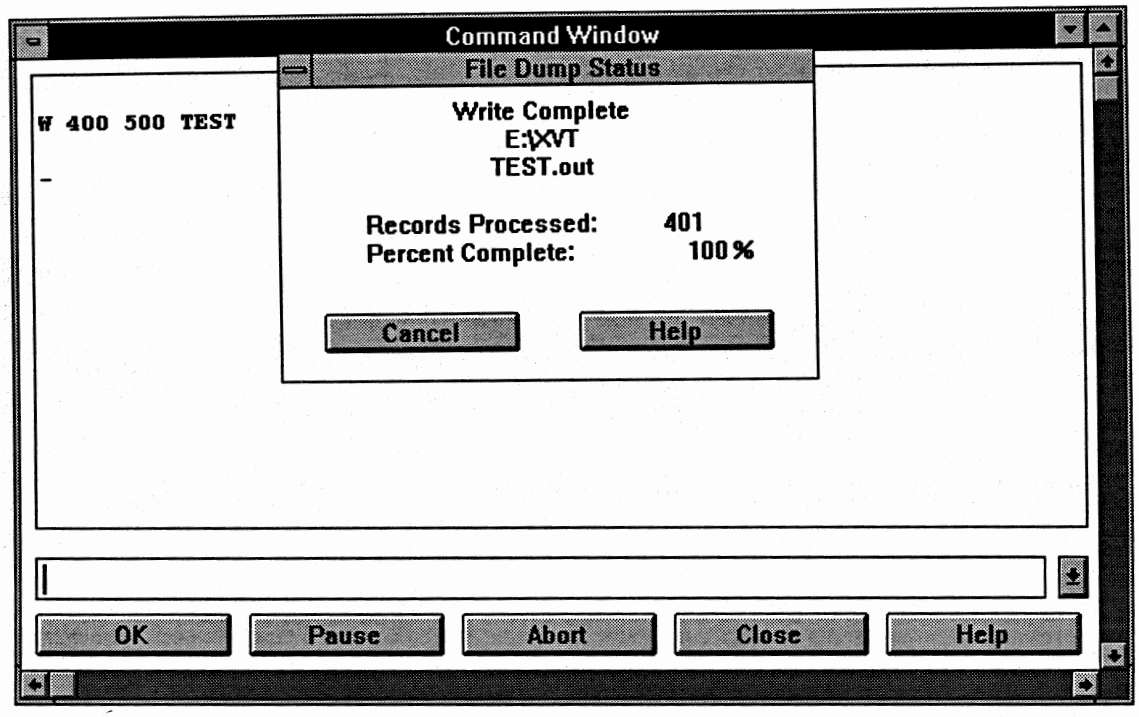
See Chapter 4 - *Using The VASM Window* for additional information on using the VASM window.

VER (Display Firmware Version) Command



This command displays the current version and release date of the BMD firmware.

W (Write Memory Contents To File) Command



This command is used to transfer files from the target system memory to a host computer. The size of the block being transferred is specified by the two address fields activated by this command. When uploading a file using the SourceGate II BMD, the block will be written as a Motorola S-Record file with symbol information added to the beginning of the file. This symbol information is in a format that is easily read by HMI software. The extent of this file defaults to .HEX. This symbol/hex file is written to disk on the host computer under the specified filename. **Note that you can also use the 'File/Upload Hex File' menu item to write a file.**

Once the Write Command has been executed, a File Load Status window will appear which will show the status of the write operation. The write operation can be aborted by pressing the **Cancel** button located in the File Load Status window.

The format for writing to the main RS232 port is:

W([SUPD]) Baddr Eaddr Filename (Flags)

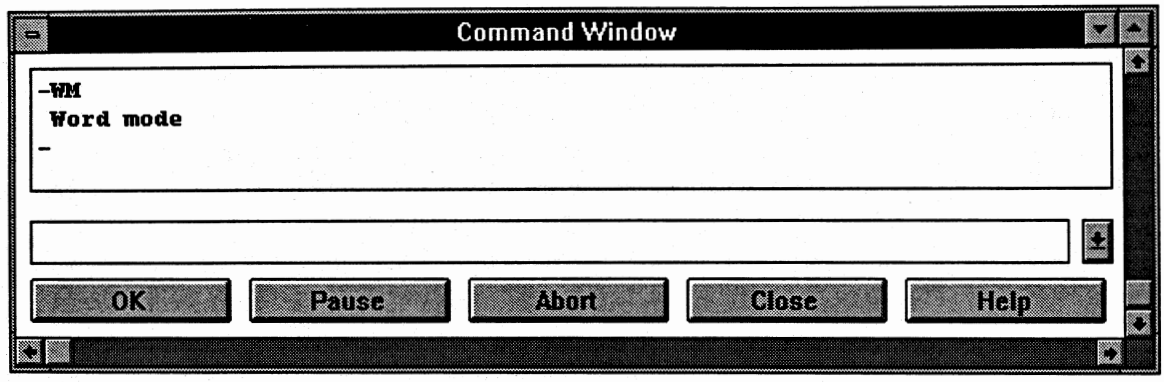
Where:

SUPD:	are the available overrides
Baddr:	is the hexadecimal beginning address of code to be written
Eaddr:	is the hexadecimal ending address of code to be written
Filename:	is a filename, valid to the host computer, that designates the file to be uploaded (written) to the host computer. If no extension is specified, the default .HEX will be used.
Flags:	are options to be used for each write The options are defined as follows:
H:	Upload/Download only the hex data records present in a hex file. Ignore all symbol records.
S:	Upload/Download only the symbol records of a hex file. Ignore all hex data records.

The 'W' command opens the given file for writing, and writes the contents of the SourceGate internal symbol table into this file, along with the data bytes in memory, from Baddr through Eaddr. 'Baddr' and 'Eaddr' are hexadecimal numbers (or symbols).

Note: The default file extension used on read (load) and write (dump) operations can be changed by modifying the .SG2 file for your BMD. For read operations, the "DefaultLoadExtension" definition can be changed to reflect the desired extension. For write operations, change "DefaultDumpExtension".

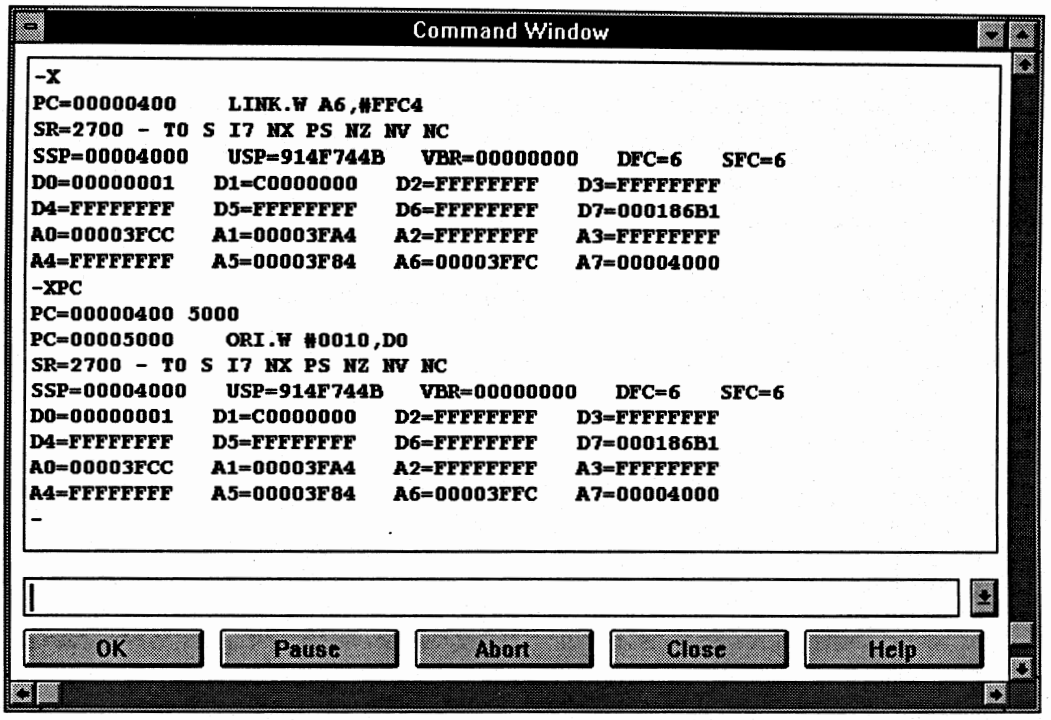
WM (Word Display Mode) Command



This command sets the default for input or output values to "Word Mode". While in Word mode, commands such as D (Dump memory) will display values in Word mode instead of long word mode or byte mode, which are the other choices (see LM and BM).

In addition, commands which require an input value will be affected by this command. For example, if the Fill command, F 8000 9000 33 is executed while in Word mode, each word of memory from 8000H through 9000H will contain 0033H. If this same command was executed with the BMD in long word mode, then 00000033H would be stored in each long word location from 8000H to 9000H. If this same command was executed with the BMD in byte mode, then 33H would be stored in each byte location from 8000H to 9000H.

X (Examine/Change Registers) Command



This command allows you to examine and modify the registers and flags (status register). There are three options to the examine command. The first option is to display all registers and flags. The second option is to display a specific register and allow it to be modified. The last option is to allow the flags to be modified by typing in the condition code status. **Note that you can also use the Register window to display and change the register contents.**

The format for displaying the registers and condition codes is:

X

The format for displaying a specific register and allowing it to be modified is:

XReg

The format for displaying just the condition code values is:

XFL

Where:

Reg: is any processor register such as A1 or SR

When changing a register using the "Xreg" command, you will be shown the current value of the register. To change it, simply type in the new value and then press the "ENTER" key. The register will then contain the new value.

When changing a condition code value using the "XFL" command, you will be shown the current value of the status register and corresponding condition codes. To change condition codes, enter the codes to be changed and then press the "ENTER" key. The register display will then show the modified condition codes.

An explanation of each of the condition codes is listed below:

SR Bit	Flag or Condition Code Name	Not Active Code	Active Code
15	Trace Enable T1	T0 or T1	T2 or T3
14	Trace Enable T0	T0 or T2	T1 or T3
13	Super/User State	U	S
12	Mast/Int State	I	M
11	Not Defined	--	--
10	Int Priority Mask 2		
9	Int Priority Mask 1	I0 through I7	
8	Int Priority Mask 0		
7	Not Defined	--	--
6	Not Defined	--	--
5	Not Defined	--	--
4	Extend	NX	XT
3	Negative	PS	NG
2	Zero	NZ	ZR
1	Overflow	NV	OV
0	Carry	NC	CY

Use these codes, after entering the "XFL" command, to change to a different condition code. For example, if you perform an "XFL" command and are shown the following:

SR = 2709 NT S I7 NX NG NZ NV CY

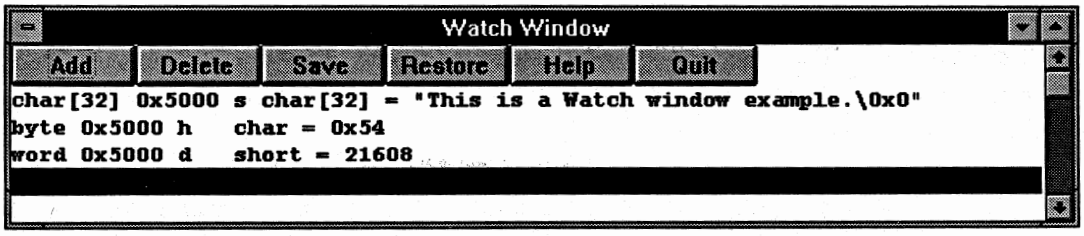
then you enter "U I6 ZR" and press the "ENTER" key, you will see the following:

SR = 060D NT U I6 NX NG ZR NV CY

Working With The Watch Window

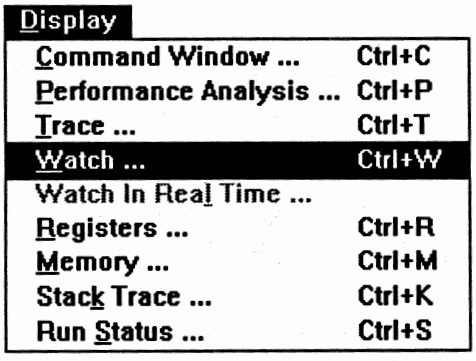
The Watch window allows you to "watch" certain memory locations or variables. You can choose the variable type, address, and display type that is shown in the Watch window. Data structure information along with the display of local variables is supported when Archimedes, Software Development Systems (SDS), or any other compiler that can produce IEEE-695 or Common Object File Format (COFF) output files is used. Watch windows can be saved to disk and loaded again at a later date. Most variable types are supported and can be displayed.

The Watch window is updated each time emulation is terminated through a breakpoint, stop command, or reset command. It will also be updated during single step operations. An example Watch window is shown below:



Opening The Watch Window

To open the Watch window, choose 'Display/Watch':



An alternate method of opening the Watch window is to use the accelerator key combination Ctrl+W (Windows version only).

Adding A Variable To The Watch Window

Variables can be added to the Watch window by clicking on the **Add** button or by entering in the variable information manually. Each of these methods will be reviewed in the following sections.

Entering Watch Variables Manually From The Watch Window

The Watch window allows the user to enter watch variables manually by clicking on a line in the watch window and typing in the desired type, address, and display type information for the variable to be "watched" in the format of:

type address display_type

where 'type' is the variable type, 'address' is the address (or symbol name) of the variable, and 'display_type' is the format you wish the variable displayed in. A space must be used between each of these entries. After entering in the desired information, press the 'Enter' key to store the entry into the Watch window. The current value of the newly entered watch variable will then be displayed.

Probably the easiest way to enter a watch variable that is a symbol name is to press the right mouse button while in the Watch window. This will invoke the Symbol Selection window from which a symbol can be chosen by clicking on the desired variable and then pressing the OK button. Once the symbol has been selected, it will automatically be added to the watch variable list.

Each of the manual entry fields (type, address, display_type) are described in the sections that follow.

Type

The type field is used to tell the Watch window what type of variable is being specified. Any type can include an '*' after the type to denote a pointer.

The type field is used to tell the Watch window what type of variable is being specified. Any type can include an '*' after the type to denote a pointer. Some examples follow:

char	Variable at the address is of type char
char*	Value is a pointer to a variable of type char
char**	Value is a pointer to a pointer that points to a variable of type char

The following types are available and should be entered as shown:

byte	double
char	unsigned byte
int (or integer)	unsigned char
short	unsigned int (or integer)
word	unsigned short
long	unsigned word
address	unsigned long
float	

Address

The address field is used to tell the Watch window the address of the data to be watched. This field can be entered in decimal (default) or in hexadecimal by preceding the address with "0x" or "0X". On some emulators, you can specify an override designator as part of an address. Overrides may precede the address in the form {override}address. For example, if you wanted to watch an integer in the supervisor/data space of a 68000 based system at location 5000 hex and have it displayed as a decimal value, you should enter "integer {SD}0x5000 d".

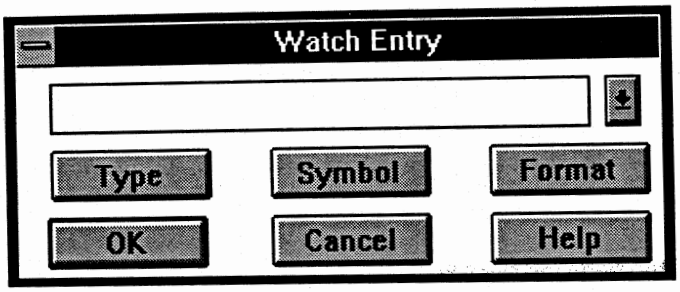
Display_Type

The display type field is used by the Watch window to determine the way in which the data will be displayed. The default display type is decimal. All types must be preceeded with "%", the case is ignored. An example of the available display types follows:

%h (or x)(Hex)
%d (Decimal)
%o (Octal)
%b (Binary)
%c (Char)
%s (String)
%u (Unsigned Decimal)
%f (Float)

Using The Watch Entry Window To Enter Watch Variables

To add a variable to the Watch window using the 'Watch Entry' window, press the 'Add' button. This will open the 'Watch Entry' window:



While in this window, you can enter a watch variable by using the buttons shown in this window or by typing in the information manually. The following sections explain how to enter a variable using the buttons and manually.

Adding A Watch Variable Manually

The 'Watch Entry' window allows you to enter watch variables manually. From here, you can enter the type, address, and display type information for the variable to be "watched" in the format:

type address display_type

where 'type' is the variable type, 'address' is the address (or symbol name) of the variable, and 'display_type' is the format you wish the variable displayed in. A space must be used between each of these entries. If you realize you have made a mistake after you have chosen OK, you can edit any previous entry by pressing the **Scroll Box** button located next to the variable entry field. This allows you to load a previous entry into the 'Watch Entry' window.

If the variable to be watched is a pre-defined symbol, you can bypass the above steps by typing the name of the symbol in the watch entry field and pressing the OK button. SourceGate will search the symbol table information and display the variable in its proper form.

Probably the easiest way to enter a watch variable that is a symbol name is to press the right mouse button while in the Watch window. This will invoke the Symbol Selection window from which a symbol can be chosen by clicking on the desired variable and then pressing the OK button. Once the symbol has been selected, it will automatically be added to the watch variable list.

Each of the manual entry fields (type, address, display_type) are described in the sections that follow.

Type
The type field is used to tell the Watch window what type of variable is being specified. Any type can include an '*' after the type to denote a pointer.

Some examples follow:

char	Variable at the address is of type char
char*	Value is a pointer to a variable of type char
char**	Value is a pointer to a pointer that points to a variable of type char

The following types are available and should be entered as shown:

byte	double
char	unsigned byte
int (or integer)	unsigned char
short	unsigned int (or integer)
word	unsigned short
long	unsigned word
address	unsigned long
float	

Address
The address field is used to tell the Watch window the address of the data to be watched. This field can be entered in decimal (default) or in hexadecimal by preceding the address with "0x" or "0X". On some emulators, you can specify an override designator as part of an address. Overrides may precede the address in the form {override}address. For example, if you wanted to watch an integer in the supervisor/data space of a 68000 based system at location 5000 hex and have it displayed as a decimal value, you should enter "integer {SD}0x5000 d".

Display_Type
The display type field is used by the Watch window to determine the way in which the data will be displayed. The default display type is decimal. All types must be preceeded with "%", the case is ignored. An example of the available display types follows:

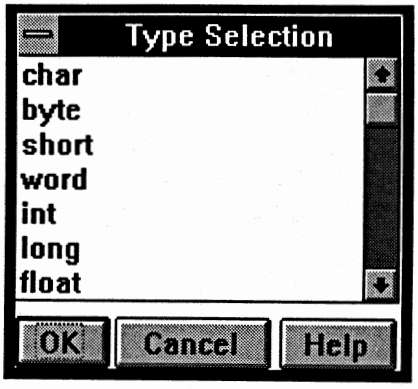
%h (or x)(Hex)
%d (Decimal)
%o (Octal)
%b (Binary)
%c (Char)
%s (String)
%u (Unsigned Decimal)
%f (Float)

Adding A Watch Variable Using Buttons

The Watch Entry window buttons can be used to enter the information for a watch variable definition. The **Type**, **Symbol**, and **Format** buttons invoke additional windows from which watch variable parameters can be defined. An explanation on how to use each of these buttons is defined in the sections that follow. **It should be noted that, if you are using buttons to enter watch variables, you should not enter any parameters manually until all of the desired buttons have been pressed. If you previously entered information manually and then press a button, the manually entered information will be deleted. To mix manual and button entries, always use the buttons first and then edit the watch entry line manually.**

Type

The **Type** button will open the 'Type Selection' window as shown below:



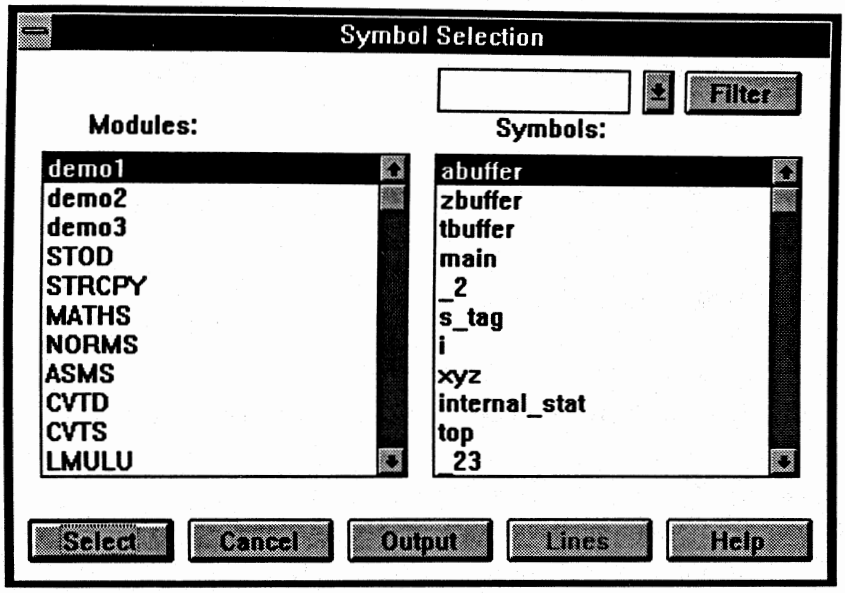
From this window, you can choose most of the available variable types. Additional types can be entered manually. To choose a type, click on it and then press the **Ok** button. The selected type will then be shown in the 'Watch Entry' window. The pre-defined choices are as follows:

char
byte
short
word
int
long
float
double
address
char *
int *
long *

If you decide not to enter a type, press the **Cancel** button. To obtain help for this window, press the **Help** button.

Symbol

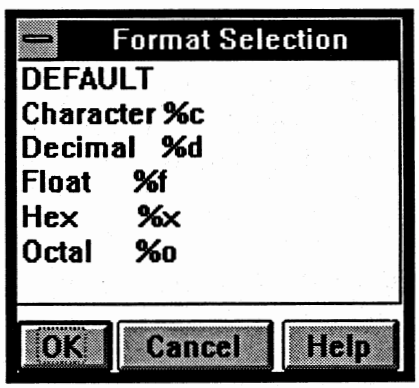
The **Symbol** button will open the 'Symbol Selection' window as shown below:



From this window, you can select one of the pre-defined symbols as the variable to be watched by highlighting it and pressing the **Select** button. The selected symbol will then appear in the 'Watch Entry' window. For additional information on using this window, see ***Working With Symbols*** in Chapter 2.

Format

The **Format** button will open the 'Format Selection' window as shown below:



The Format Selection window allows you to choose a pre-defined variable display format by using the mouse pointer and left mouse button to highlight the desired format. Pressing the **Ok** button will load the defined format into the 'Watch Entry' window. The output value for this watch variable will be in the specified format when displayed in the Watch window. Note that DEFAULT values are set to the most common format for the defined type (e.g. pointers will default to hex, characters to ASCII, integers to decimal, etc.). If you decide not to enter a display type, press the **Cancel** button. To obtain help for this window, press the **Help** button.

Ok
The **Ok** button will accept the defined watch entry and place the variable in the Watch window. If there are any formatting errors encountered with the entry, they will be displayed in the Watch window. To edit a previously entered watch entry, press the **Scroll Box** button located next to the watch entry field and choose one of the entries shown. You can then edit it.

Cancel
The **Cancel** button will cancel the watch entry operation and will close the 'Watch Entry' window.

Help
The **Help** button will invoke help for this window.

Scroll Box
The **Scroll Box** button will allow you to choose a previously entered watch entry.

Deleting A Variable From The Watch Window
To delete a previously entered watch variable from the Watch window, highlight the variable by clicking on it, then press the **Delete** button. The variable will then be removed from the Watch window.

Saving A Watch Window
To save a Watch window's contents for future reference, press the **Save** button. Then enter the name of the file to save the Watch window to. Once this is done, the variables defined in the current Watch window will be saved to the named file. To re-load this Watch window at a later date, press the **Restore** button and choose the name of the file that you saved. The variables saved in this file will then be displayed with their current values.

Restoring A Saved Watch Window
Once you use the **Save** button to save a Watch window to a file, you can restore this saved Watch window by pressing the **Restore** button and then choosing the filename of the saved file. The variables saved in this file will then be displayed with their current values.

Watch Window Buttons
A description of the functions of the buttons available in this window are as follows:

Add
The **Add** button will add a watch variable to the Watch window.

Delete
The **Delete** button will delete a watch variable from the Watch window. If the highlight line is on the blank line immediately after the last watch entry, pressing this button will delete the last watch entry. This is handy if you wish to delete multiple watch entries. You can also use the Ctrl-Shift buttons, along with the mouse, to highlight multiple watch entries and then delete them all by pressing the **Delete** button.

Save

The **Save** button will save a Watch window's contents to a file.

Restore

The **Restore** button will restore a previously saved Watch window's contents to the Watch window.

Close

The **Close** button will close the Watch window.

Help

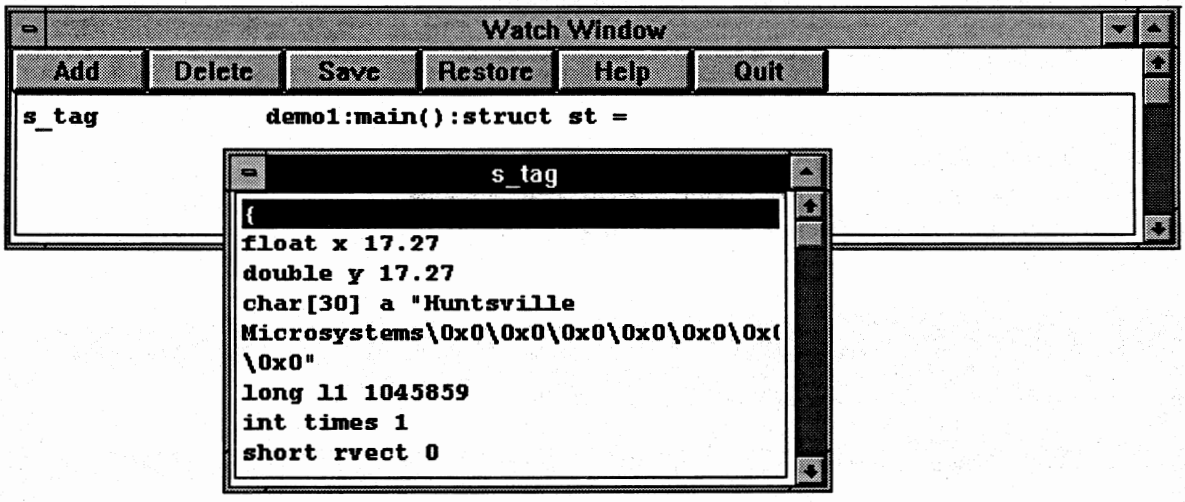
The **Help** button will invoke help for this window.

Watch Window Examples

The following examples were included in order to help give you a better understanding of how to use the Watch window.

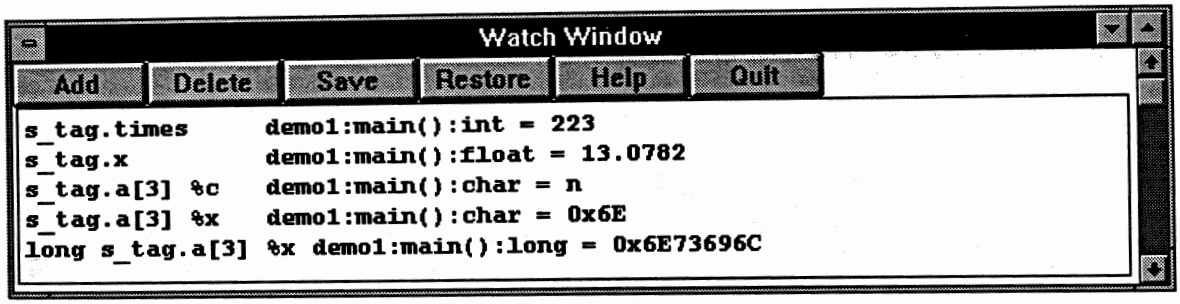
Watching A Structure

To watch an entire structure, enter the name of the structure as a watch variable. A separate window showing the contents of each element of the structure will be shown if the structure's information is too long to be shown in the Watch window:



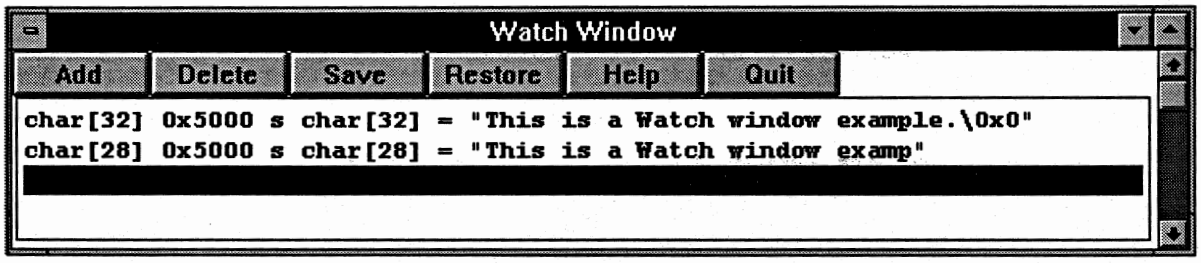
Watching An Element Of A Structure

If you wish to watch certain elements of a structure, this can be accomplished by entering "structure.element":



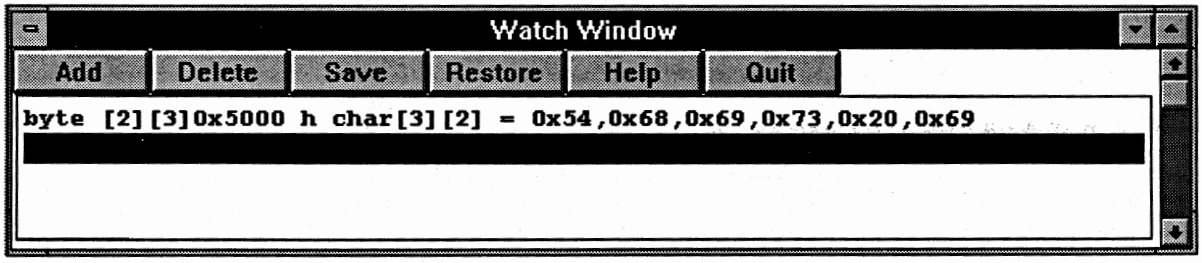
Watching A One-Dimensional Array

To watch a one-dimensional array, enter the array size. If you wish to watch certain elements of an array, this can be accomplished by entering "array_name[element]". The following is an example of a character array:



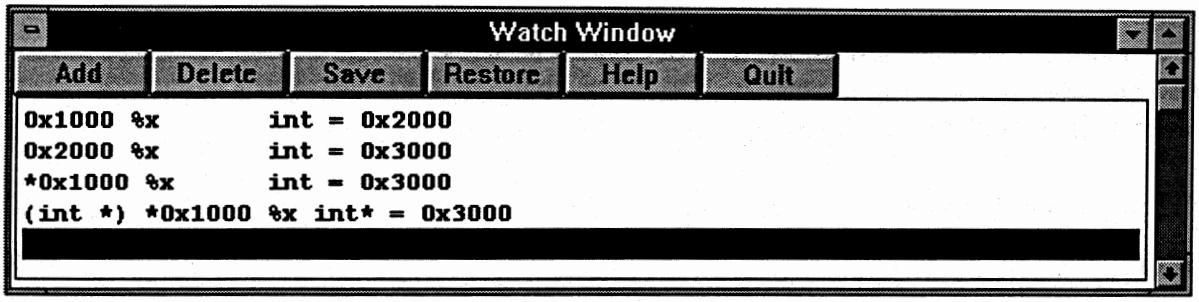
Watching A Multi-Dimensional Array

For multi-dimensional arrays, type "array_name[a][b]" to watch particular elements. Arrays will be displayed in a separate "Watch Output" window if the output if too large to be shown in the Watch window.



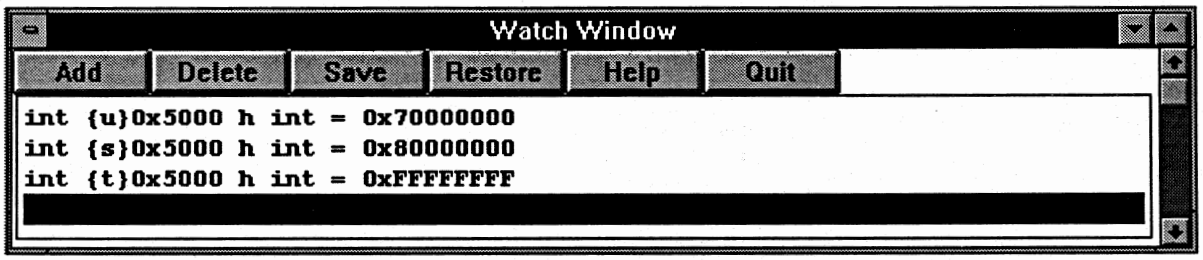
Watching Pointers

Pointers can be watched by preceeding an address with the pointer designator "*".



Using Overrides In The Watch Window

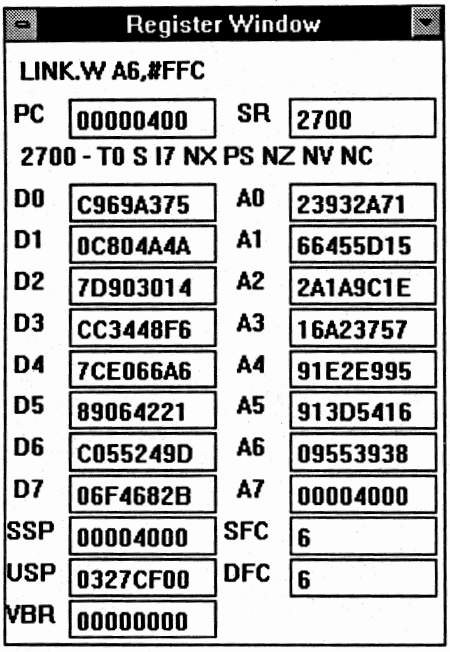
Overrides can be used to watch different memory spaces. In the example shown below, address 0x5000 in a 68030 system is being watched. The 68030 has separate Supervisor and User memory spaces. The {s} override is a 68030 override which specifies Supervisor memory space and the {u} override is a 68030 override which specifies User memory space. The {t} override specifies target memory. Notice that, using overrides, the contents of address 0x5000 is shown to be different for each memory space, which is to be expected since there are actually three distinct 0x5000 addresses. Without the use of overrides, only one space could be shown at a time.



Working With The Register Window

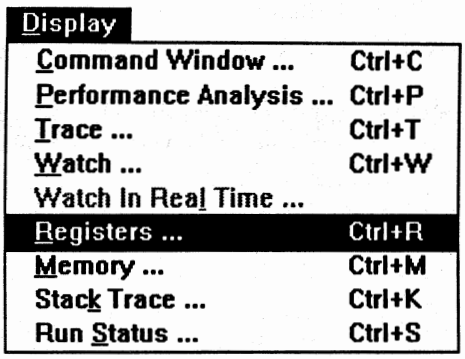
The Register window shows the contents of the processor's registers including the status of the flags and the current disassembly mnemonic being pointed to by the program counter register. The register values are always accurate whenever emulation is not in progress, and are the last known value whenever emulation is in progress. Registers can be altered in the register display window by selecting the desired register data field and then and entering a new value.

The Register window is automatically updated whenever the BMD is single-stepped, a breakpoint occurs, a stop command is executed, or the BMD is reset. The registers shown in the Register window are dependent on the background mode processor being used. An example Register window is shown below:



Opening The Register Window

To open the Register window, choose 'Display/Registers':



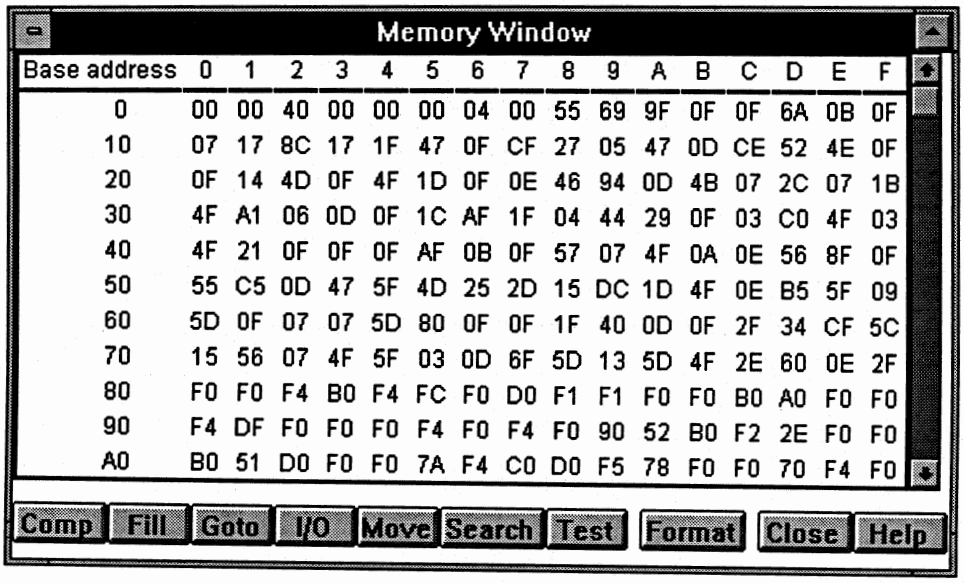
An alternate method of opening the Register window is to use the accelerator key combination Ctrl+R (Windows version only).

Modifying Register Values

To modify a register, move the insertion bar to the register to be changed and then press the left mouse button. You can then enter the desired information beginning at the insertion point. To delete the old contents of a register and add a completely new value, click and drag to highlight the entire contents of the register and then enter the new value.

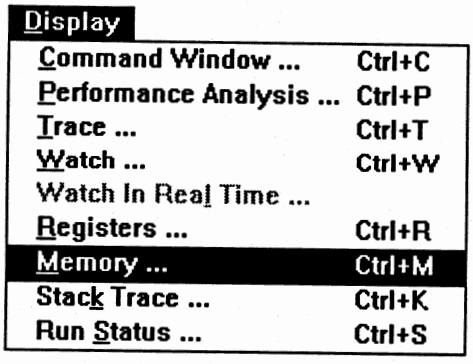
Working With The Memory Window

The memory window allows you to display and change individual memory locations. Additional features of this window allow you to compare blocks of memory, fill a memory block with a constant value, move a block of memory, search a block of memory for a particular value, and test a memory block among others. The Memory window has the following format:



Opening The Memory Window

To open the Memory window, choose 'Display/Memory':



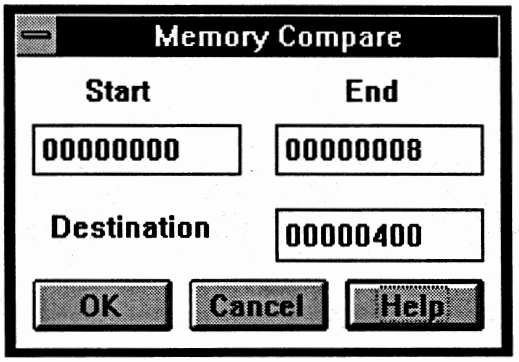
An alternate method of opening the Memory window is to use the accelerator key combination Ctrl+M (Windows version only).

Modifying Memory Values

To change a memory value, place the cursor on the value to be changed and press the left mouse button. To select the entire number, press the left mouse button and then drag until the whole number is highlighted. A new value can then be entered. After the desired value has been entered, click on another location or press one of the arrow keys to go to an adjacent memory location. This will load the value you just changed into memory at the specified address. Any values that you have changed will be shown in a different color for easy identification.

Comparing Memory Blocks

The Memory Compare window is invoked when the **Comp** button is pressed from within the Memory window. This window allows you to compare two memory blocks to see any differences between them. The Memory Compare window has the following format:



The 'Start' field defines the starting address of the first block, the 'End' field defines the ending address of the first block, and the 'Destination' field defines the starting address of the second block. When the **Ok** button is pressed, the first memory location in the first block is compared to the first location in the second block, etc., until the defined areas have been compared.

In the example above, the first block starts at address 0 and ends at address 8H. The second block begins at address 400H. When the **Ok** button is pressed, the SourceGate II BMD will begin comparing each location in the first block to the corresponding location in the second block.

Memory Compare Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will begin the memory compare operation.

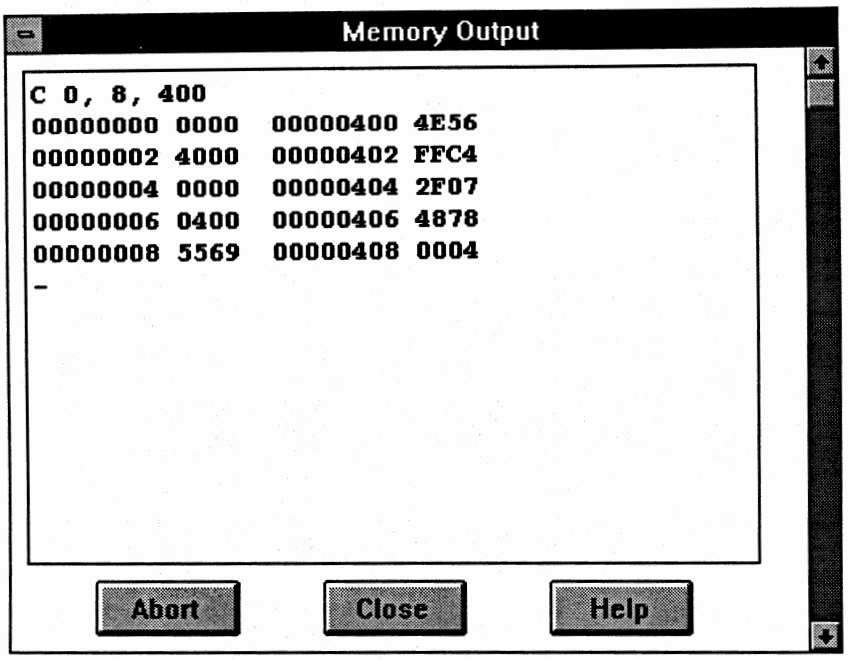
Cancel

The **Cancel** button will cancel the compare operation and will close the Memory Compare window.

Help

The **Help** button will invoke help for this window.

The results of the memory compare operation are shown in the Memory Output window. The Memory Output window has the following format:



The output of this window is of the format:

Addr1 Value1 Addr2 Value2

where:

Addr1 - is the address in block 1
Value1 - is the contents of Addr1
Addr2 - is the address in block 2
Value2 - is the contents of Addr2

If information in this format is shown, the comparison failed at these locations. If no information is shown, then all locations compared contain the same information.

Memory Output Window Buttons

A description of the functions of the buttons available in this window are as follows:

Abort

The **Abort** button will stop the output operation being performed. This could be useful if you compared a large memory block and wish to stop scrolling through the output.

Close

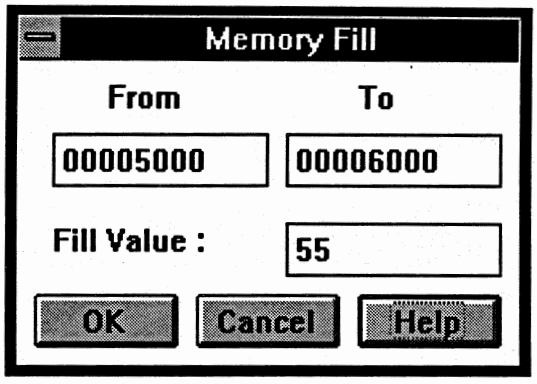
The **Close** button closes the Memory Output window.

Help

The **Help** button invokes help for this window.

Filling Memory With A Constant

The Memory Fill window is invoked when the **Fill** button is pressed from within the Memory window. This window allows you to fill a memory block with a constant value. The Memory Fill window has the following format:



The 'From' field defines the beginning address in memory where the fill will begin, the 'To' field defines the ending address in memory where the fill will stop, and the 'Fill Value' field defines the constant value that this memory block will be filled with. When the **Ok** button is pressed, the fill process will begin.

In the example above, memory from 5000H to 6000H will be filled with the value 55H when the **Ok** button is pressed.

Memory Fill Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will begin the fill operation.

Cancel

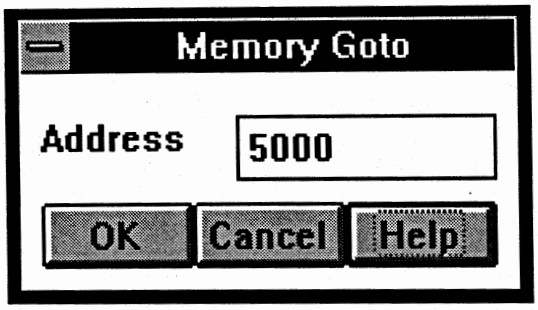
The **Cancel** button will cancel the fill operation and will close the Memory Fill window.

Help

The **Help** button will invoke help for this window.

Viewing A Specific Address In Memory

The Memory Goto window is invoked when the **Goto** is pressed from within the Memory window. This window allows you to define the base address of the memory locations shown in the Memory window. The Memory Goto window has the following format:



The 'Address' field defines the base address of the memory locations that will be shown in the Memory window. When the **Ok** button is pressed, the Memory window will be re-configured to show the memory locations beginning with the address defined here.

In the example above, the Memory window will show memory addresses starting at 5000H when the **Ok** button is pressed.

Memory Goto Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will make the Memory Window begin with the defined address.

Cancel

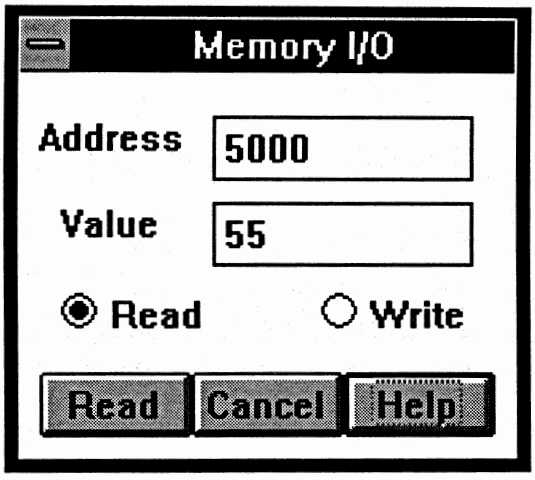
The **Cancel** button will cancel the goto operation and will close the Memory Goto window.

Help

The **Help** button will invoke help for this window.

Reading/Writing A Specific Memory Location

The Memory I/O window is invoked when the **I/O** button is pressed from within the Memory window. This window allows you to read or write a specific memory location. The Memory I/O window has the following format:



The 'Address' field defines the address to be read or written, the 'Value' field defines the value to be read or written, and the 'Read' and 'Write' check boxes define which operation is to be performed on the defined address.

In the example above, the value 55H has been read from address 5000H. This is the result of pressing the **Read** button. If a write operation was being performed, the **Read** button would change to **Write** and pressing the **Write** button would write 55H to the address 5000H.

Note that some background mode processors may not support this feature. If the I/O button is grayed in your Memory window, this feature is not available to you.

Memory I/O Window Buttons

A description of the functions of the buttons available in this window are as follows:

Read/Write

The **Read** button will read the contents of the defined address and shown the result in the 'Value' field. If a **Write** operation is selected, the **Read** button will change to **Write** and pressing this button will write the value defined in the 'Value' field to the defined address.

Cancel

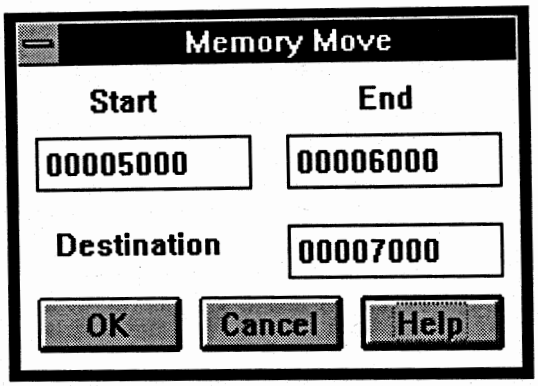
The **Cancel** button will cancel the I/O operation and will close the Memory I/O window.

Help

The **Help** button will invoke help for this window.

Moving Memory Blocks

The Memory Move window is invoked when the **Move** button is pressed from within the Memory window. This window allows you to move a block of memory to another location in memory. The Memory Move window has the following format:



The 'Start' field defines the starting address of the block to be moved, the 'End' field defines the ending address of the block to be moved, and the 'Destination' field defines the location in memory where the block is to be moved to. When the **Ok** button is pressed, the contents of the first memory location in the block to be moved is moved to the location specified in the 'Destination' field, the contents of the next location are moved to the location specified in the 'Destination' field plus one, etc., until the entire block has been moved.

In the example above, the block to be moved starts at address 5000H and ends at address 6000H. The destination for the block is 7000H. When the **Ok** button is pressed, the SourceGate II BMD will begin moving each location in the block to be moved to the corresponding location in the destination block.

Memory Move Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will begin the memory move operation.

Cancel

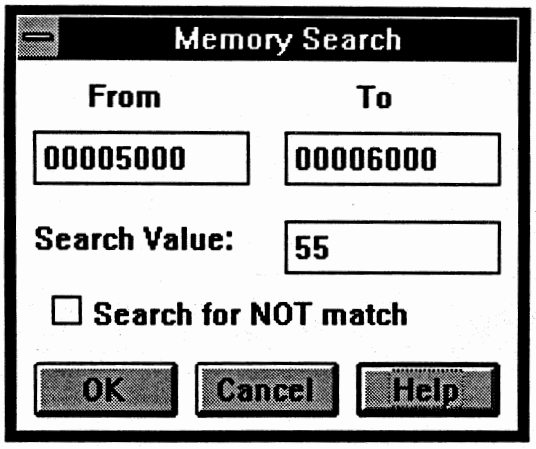
The **Cancel** button will cancel the move operation and will close the Memory Move window.

Help

The **Help** button will invoke help for this window.

Searching Memory Blocks

The Memory Search window is invoked when the **Search** button is pressed from within the Memory window. This window allows you to search a block of memory for a particular value or you can search a block of memory and display all locations that do not contain a specific value. This feature is useful for checking memory that is being initialized to a specific value. The Memory Search window has the following format:



The 'From' field is the beginning address where the search will begin, the 'To' field is the ending address where the search will end, and the 'Search Value' is the value to be searched for. The 'Search for NOT match' check box should be used to search for any occurrences in the defined block that do NOT contain the value defined in the 'Search Value' field.

In the example above, the block to be searched begins at location 5000H and ends at location 6000H. The value to be searched for is 55H. When the **Ok** button is pressed, the SourceGate II BMD will begin searching the specified block for the defined value.

Memory Search Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will begin the memory search operation.

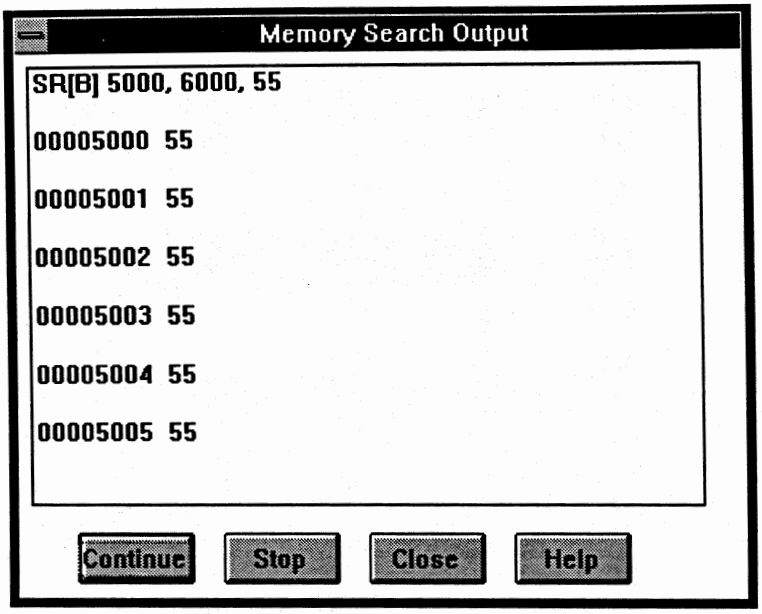
Cancel

The **Cancel** button will cancel the search operation and will close the Memory Search window.

Help

The **Help** button will invoke help for this window.

The results of the memory search operation are shown in the Memory Search Output window. The Memory Search Output window has the following format:



The output of this window is of the format:

Addr Value

where:

Addr - is the address of the search match
Value - is the value located at the address of the search match

If no information is shown, the search found no matches.

Memory Search Output Window Buttons

A description of the functions of the buttons available in this window are as follows:

Continue

Once a search match is found, searching will stop and the Memory Search Output window will shown the found location. The **Continue** button will allow you to begin searching for the next match.

Stop

The **Stop** button will stop the search operation.

Close

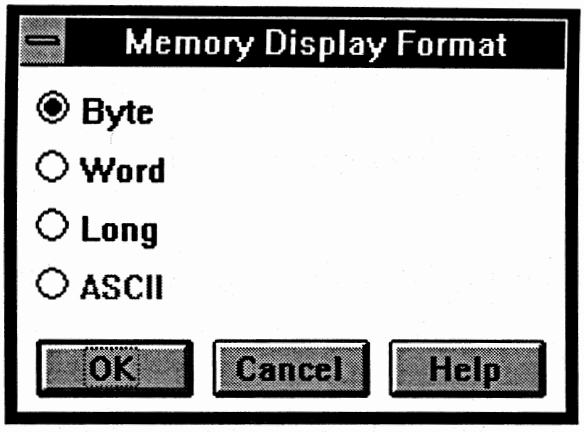
The **Close** button will close the Memory Search Output window.

Help

The **Help** button will invoke help for this window.

Specifying The Display Format

The Memory Display Format window is invoked when the **Format** button is pressed from within the Memory window. This window allows you to change the display format of the Memory window. The Memory Display Format window has the following format:



To select a display format, you should place the mouse pointer on the desired format and then press the left mouse button. Once this is done and the **Ok** button is pressed, the Memory window will show memory values in the selected display format.

Memory Display Format Window Buttons

A description of the functions of the buttons available in this window are as follows:

Ok

The **Ok** button will change the Memory window display to the selected format.

Cancel

The **Cancel** button will cancel the format change operation and will close the Memory Display Format window.

Help

The **Help** button will invoke help for this window.

Memory Window Buttons

A description of the functions of the buttons available in the Memory window are as follows:

Comp

The **Comp** button will allow you to compare two memory blocks and display any differences between them.

Fill

The **Fill** button will allow you to fill memory with a constant value.

Goto

The **Goto** button will allow you to define the starting address of the memory displayed in the Memory window.

I/O

The **I/O** button will allow you to read or write a specific memory location.

Move

The **Move** button will allow you to move a block of memory to another location.

Test

The **Test** button will allow you to test RAM locations to insure that they are functioning properly.
Note: This button is not functional with the BMD.

Search

The **Search** button will allow you to search memory for a particular value.

Format

The **Format** button will allow you define the format of the values shown in the Memory Window display. The available formats are 'Byte', 'Word', and 'Long'.

Close

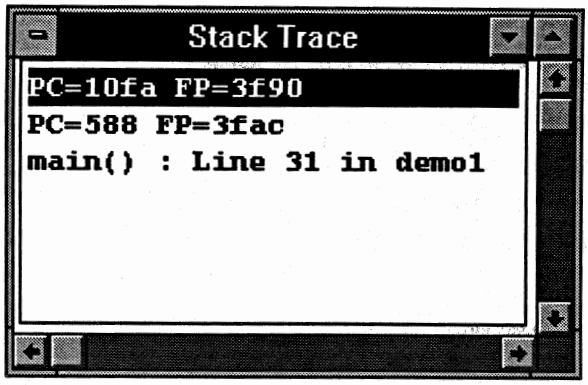
The **Close** button will close the Memory window.

Help

The **Help** button will invoke help for this window.

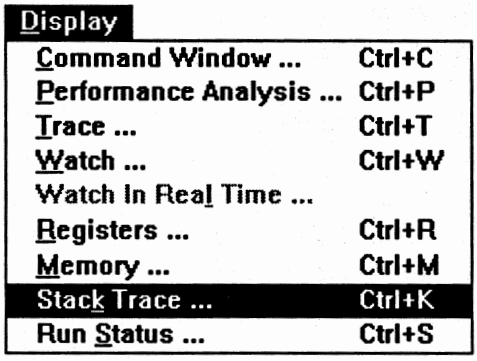
Working With The Stack Trace Window

The Stack Trace window shows the functions and local variables that are pushed onto the stack. The local variables passed and their values are displayed in their default base (hexadecimal, decimal, etc.). When stepping in assembly language, the Stack Trace window shows the current program counter and frame pointer values. The Stack Trace window has the following format:



Opening The Stack Trace Window

To open the Stack Trace window, choose 'Display/Stack Trace':



An alternate method of opening the Stack Trace window is to use the accelerator key combination Ctrl+K (Windows version only).

Stack Trace Window Contents

The Stack Trace window is automatically updated whenever the BMD is single-stepped, a breakpoint occurs, a stop command is executed, the BMD is reset, or memory is modified. The information shown in this window is the true information that is currently located on the stack. This provides you a way to track what is currently on the stack.

The information shown may be a combination of source line addresses and assembly addresses. The source line information will be in the format:

function (type var=value) : Line x in module

where:

function :	is the calling function
type:	is the variable type being passed
var:	is the variable name being passed
value:	is the value of the variable being passed
x:	is the line number of the calling module
module:	is the calling module

The assembly information will be in the format:

PC=X FP=Y

where:

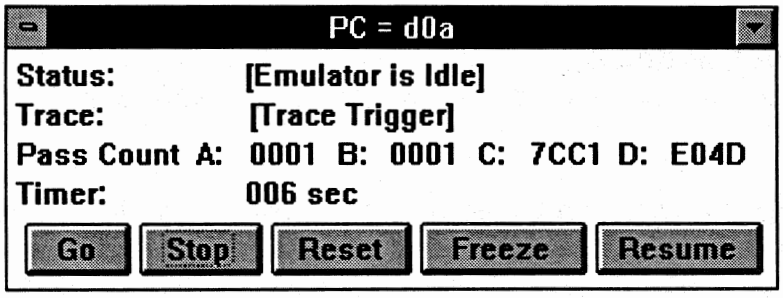
PC:	is the program counter
X :	is the value of the program counter
FP:	is the frame pointer
Y:	is the value of the frame pointer

An additional feature available from this window is the ability to call up a source or assembly window by double clicking on a line shown in the Stack Trace window or by pressing ENTER while the desired Stack Trace entry is highlighted. Once this is done, the source or assembly module associated with that Stack Trace entry is shown in a source or assembly window.

This version of SourceGate will display local variable values in this window for Archimedes and Software Development Systems (SDS) compilers and any compiler that can produce Common Object File Format (COFF) or IEEE-695 output files. Future releases may provide this feature for other compilers.

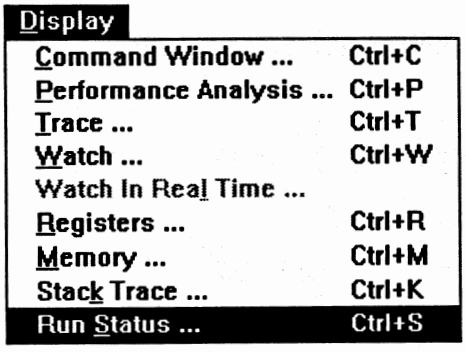
Working With The Run Status Window

The Run Status window allows you to view the emulation status using one window. The information shown in the border of this window includes the current program counter information or, if emulation is in progress, "RUNNING" is displayed. Inside of this window, the current emulation status (running or idle) is displayed. In addition, there are buttons available that allow you to begin or stop emulation and reset the processor. The Run Status window has the following format:



Opening The Run Status Window

To open the Run Status window, choose 'Display/Run Status':



An alternate method of opening the Run Status window is to use the accelerator key combination Ctrl+S (Windows version only).

Run Status Window Fields

The Run Status window has several fields that show you the status information. These fields are information only fields and you cannot change them. An overview of these fields is given in the sections that follow:

Run Status 'Status' Field

The Run Status window 'Status' field will show the current status of the BMD. If the BMD is emulating, this field will display "Emulator is running". If the BMD is idle, this field will display "Emulator is idle".

Run Status Trace Field
This field is not functional when using the BMD.

Run Status Pass Count Fields
This field is not functional when using the BMD.

Run Status Timer Field
This field is not functional when using the BMD.

Run Status Window Buttons
A description of the functions of the buttons available in this window are as follows:

Go
The **Go** button begins emulation from the current program counter location shown in the border of the Run Status window.

Stop
The **Stop** button stops emulation.

Reset
The **Reset** button resets the BMD to its defined reset location.

Freeze
This button is not functional when using the BMD.

Resume
This button is not functional when using the BMD.

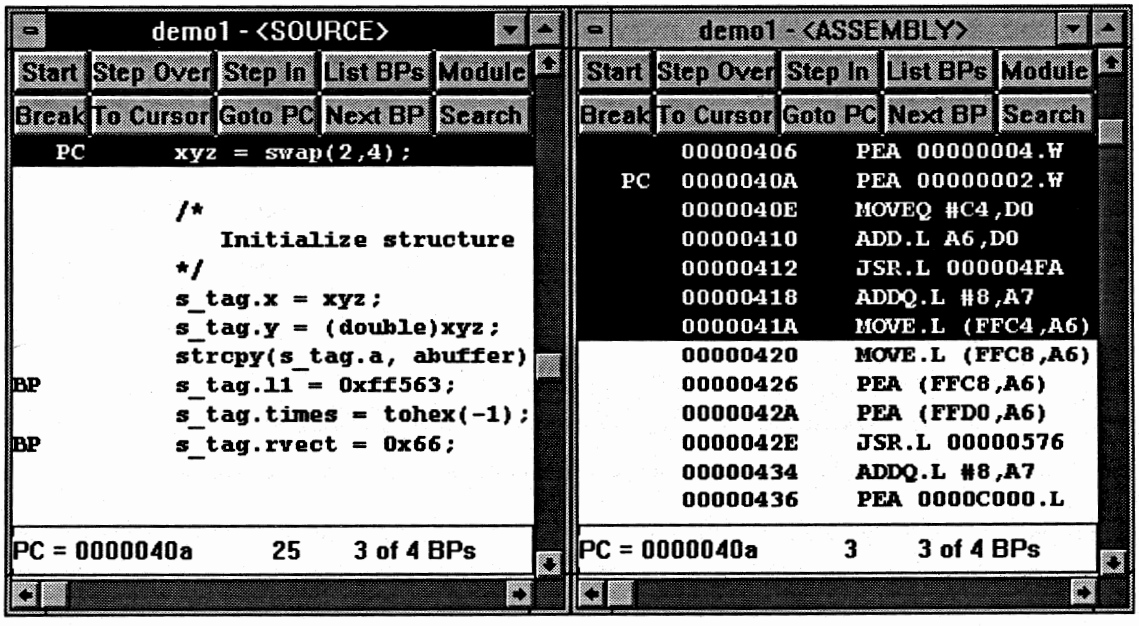
4 The Source Code Interface

When debugging source or assembly code, the Source Code Interface features of the SourceGate II BMD will prove to be invaluable to you. There are Code windows that allow you to perform BMD operations directly from your source file. There is also a Single Step window were various stepping scenarios can be performed along with a Go From/To window where you can define a source line to begin emulation from and source lines where emulation will stop. Each of these windows will be explained in detail in the sections that follow.

Using The Code Window

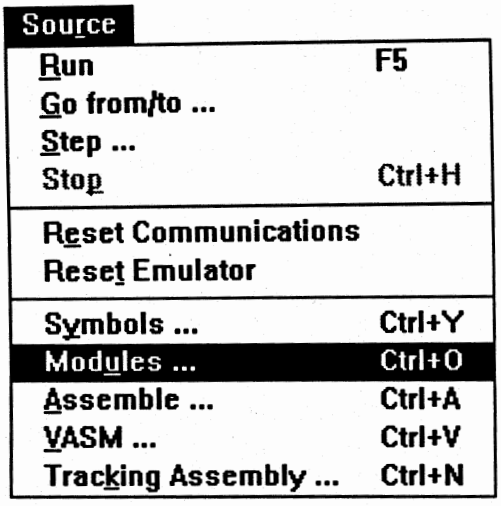
From inside of the Code windows, you can start and stop emulation, set breakpoints in your code, step over and into modules and subroutines, search your code for specific strings, jump to the current program counter location, list all set breakpoints, and open a different module to work with. Additional information is also shown, such as the number of breakpoints set, the current program counter value, and what lines of assembly are associated with the highlighted source line. Essentially, the entire operation of the background mode processor can be controlled from your source file.

An example of the Code window follows showing a source code module in one window with its corresponding assembly information shown in the assembly window next to it. Notice that the highlighted source line in the source window has the corresponding assembly for that line highlighted in the assembly window:

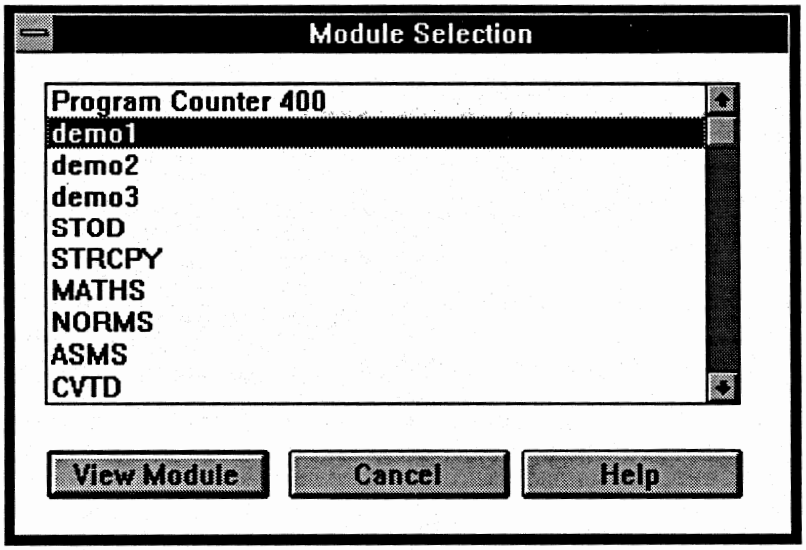


Opening A Code Window

To open a Code window, you must first load your source file using the 'File/Download Hex File' menu. Be sure to load a file that has been compiled with source level information. This information will be required to used the Source Code Interface. See your compiler documentation for additional information about creating an output file containing source level information. Once this file has been downloaded, you should choose 'Source/Modules':



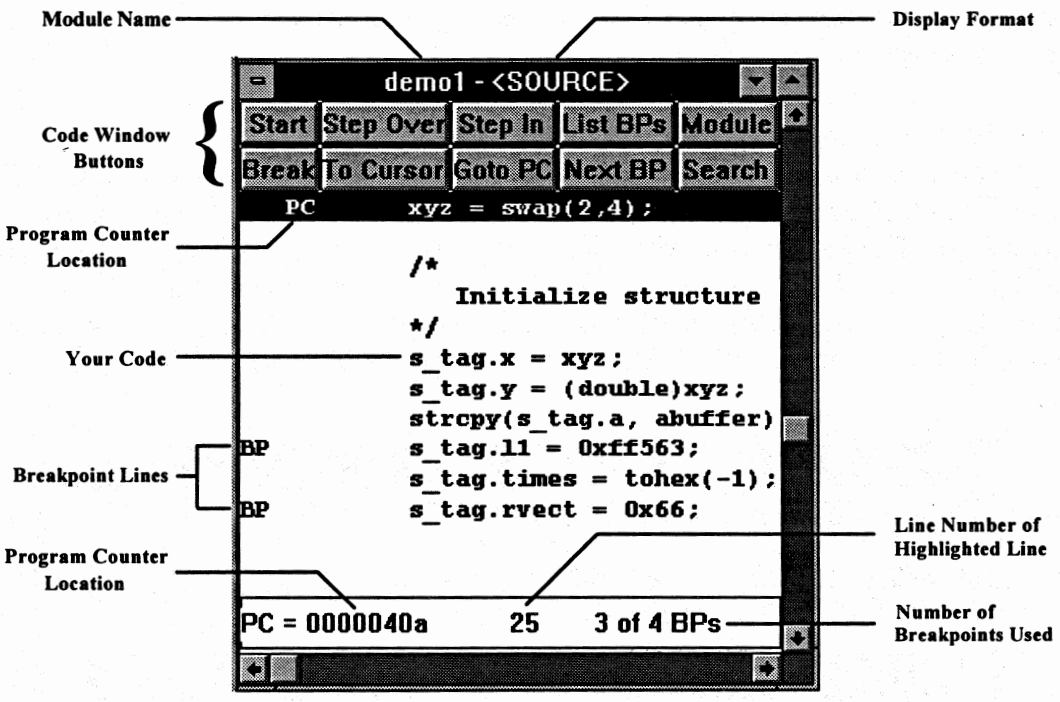
or use the accelerator key combination Ctrl+O (Windows version only). This will open the Module Selection window which allows you to view the modules that were loaded as part of your download file:



You can now highlight the module you wish to load into a Code window and then press the **View Module** button. A Code window will then be opened which contains the code for the selected module. To cancel the Module Selection operation, press the **Cancel** button. The **Help** button will invoke help for this window.

Code Window Format

The Code window was designed so that the maximum amount of information could be shown without the window becoming too cluttered. In the top border of the Code window, the module that you are viewing is shown followed by <SOURCE> or <ASSEMBLY>, depending on which display format you have selected. Directly below this information are two rows of buttons that allow you to perform certain functions. An overview of these buttons and the functions that they perform is provided in the sections that follow. The actual code information begins below the buttons. Any breakpoints that are set are designated by a "BP" label to the left of the line defined as a breakpoint. **Note: The breakpoint address must reside in RAM since the BMD will attempt to write a BGND instruction there. If the BMD is not able to write to the breakpoint address, the message "Warning: cannot install breakpoint instruction [BGND] at location xxxxxx" will be displayed where "xxxxxx" is the address of the first line of assembly code corresponding to the source instruction where the breakpoint is set.** If "PC" is shown to the left of a line, that line is where the program counter currently resides. The bottom of this window shows the current program counter (PC) value, the line number of the currently highlighted line, and status information telling how many of the four available breakpoints are set. An example of a typical Code window is shown as follows:



While you are performing stepping operations, the line containing the PC will be the line that is highlighted. This line will move as stepping proceeds into your code, opening up new source or assembly windows as needed to display where the PC is located. To select a line in the Code window, simply move the cursor to the desired line and click on it. This line will then become highlighted. Once a line has been highlighted, operations can be performed on it. These operations are reviewed in the sections that follow.

Setting Code Window Breakpoints

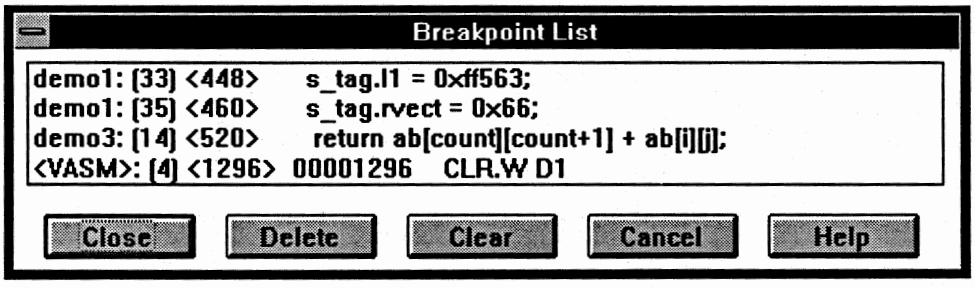
Setting a breakpoint on a line in a Code window will stop emulation when that line is encountered. Up to four breakpoints can be set in multiple Code windows. To set a breakpoint in a Code window, highlight the line you wish to set a breakpoint on by clicking on it. Then, once the line is highlighted, press the **Break** button. The highlighted line will then be labeled with "BP" showing that a breakpoint is set on that line. In addition, the bottom of all of the open Code windows will show "X of 4 BPs", where "X" is the number of breakpoints that have been set.

Note: The breakpoint address must reside in RAM since the BMD will attempt to write a BGND instruction there. If the BMD is not able to write to the breakpoint address, the message "Warning: cannot install breakpoint instruction [BGND] at location xxxxxx" will be displayed where "xxxxxx" is the address of the first line of assembly code corresponding to the source instruction where the breakpoint is set.

If you try to set more than four breakpoints, the Delete Breakpoint window will appear and you must first delete a breakpoint before your new breakpoint can be set. **Note: For SourceGate II BMD to find breakpoints set in the Code window, emulation must be started from the Code window by pressing the Start button.**

Deleting Code Window Breakpoints

Deleting breakpoints set in the Code window can be performed in one of two ways. Probably the easiest way is to highlight the line containing the breakpoint you wish to remove and then press the **Break** button. The **Break** button will toggle the breakpoint on and off for the highlighted line. The second method of deleting breakpoints in the Code window is through the use of the Breakpoint List window. This window lists all breakpoints set in the Code windows and is invoked by pressing the **List BPs** button. An example Breakpoint List window is shown as follows:



This window enables you to view the current set breakpoints and delete any of those not needed. Breakpoints are listed in the format:

module:[line#]<hex address> source line

where:

module — is the module that the breakpoint resides in. If the breakpoint is located in an assembly area that has no source line associated with it, the module will be listed as <VASM>.

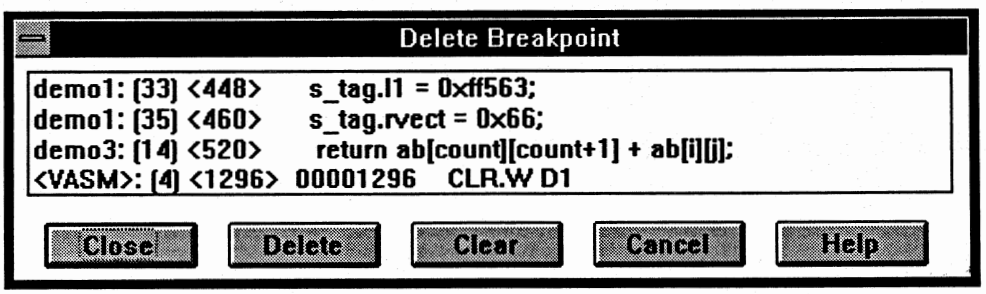
[line#] — is the line number of the module where the breakpoint is set. If <VASM> is shown as the module name, this number is the number of lines from the top of the VASM display to where the breakpoint is set.

<hex address> is the hexadecimal address of the line where the breakpoint is set.

source line — is the actual source or assembly listing of the breakpoint line.

To delete a breakpoint using this method, choose the breakpoint by clicking on it. This will highlight the selected breakpoint. The breakpoint can then be deleted by pressing the **Delete** button. All set breakpoints can be cleared by pressing the **Clear** button.

It is important to note that, if you attempt to set more than four breakpoints in the Code window, a window named Delete Breakpoint will appear:

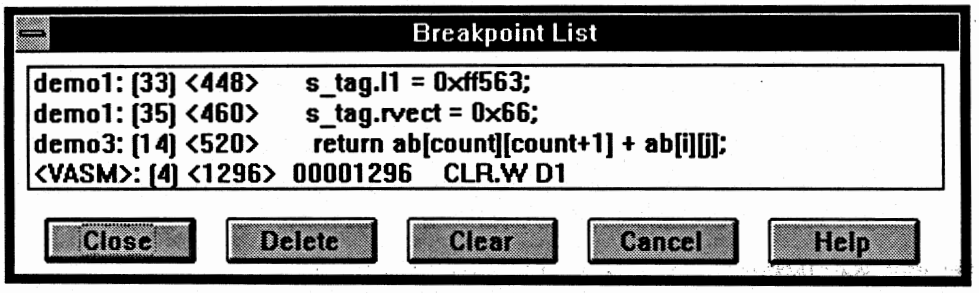


This window works exactly like the Breakpoint List window and is shown to you automatically so that you can delete a currently defined breakpoint to make room for your new breakpoint. Once you have selected/deleted a breakpoint, pressing the **Close** button will define your new breakpoint at the location you specified.

For information on the buttons available in the Breakpoint List and Delete Breakpoint windows, see ***Breakpoint List/Delete Breakpoint Window Buttons***.

Listing Code Window Breakpoints

To list the breakpoints set in a Code window, press the **List BPs** button. This will open the Breakpoint List window. This window lists all breakpoints set in the Code windows. An example Breakpoint List window is shown as follows:



This window enables you to view the current set breakpoints and delete any of those not needed. For additional information on how to use this window, see ***Deleting Code Window Breakpoints***. For information on the buttons available in the Breakpoint List window, see ***Breakpoint List/ Delete Breakpoint Window Buttons***.

Breakpoint List/Delete Breakpoint Window Buttons

A description of the functions of the buttons available in these windows are as follows:

Close

The **Close** button will close these windows. If you are using the Delete Breakpoint window and have deleted a breakpoint, pressing the **Close** button will add your new breakpoint to the list.

Delete

The **Delete** button will delete the highlighted breakpoint.

Clear

The **Clear** button will clear all currently defined breakpoints.

Cancel

The **Cancel** button will restore all breakpoints that have been deleted and will close this window.

Help

The **Help** button will invoke help for this window.

Finding Subsequent Code Window Breakpoints

To find subsequent Code window breakpoints, press the **Next BP** button. This will automatically place the highlighted line at the next line of code that is defined to be a breakpoint. This button will find breakpoints in the order that they were set. For example, if you cleared all existing breakpoints, then set breakpoints on line 5 of module A, line 8 of module B, and line 7 of module C, in that order, pressing the **Next BP** button will take you each of these breakpoints in order. However, the first breakpoint found depends on where you are when you press the **Next BP** button. If the highlighted line is on line 6 of module B when you press the **Next BP** button, the highlighted line will move to line 8 of module B (the second breakpoint set), another press of the **Next BP** button will take you to line 7 of module C (the third breakpoint set), and another press of the **Next BP** button will take you to line 5 of module A (the first breakpoint set). As you can see, the breakpoint that you go to when you press this button depends on where you start and the order in which the breakpoints were set.

Finding The Current Program Counter While In A Code Window

To find the current program counter (PC) location while in a Code window, press the **Goto PC** button. This will take the highlighted line to the Code window and line containing the PC. If the PC is not in a Code window that is open, and the PC points to a line of source code, the module containing the line pointed to by the PC will be loaded into a Code window and the line containing the PC will be highlighted. If the PC points to a line of assembly code, an assembly window will be opened beginning with the address of the PC.

Running To A Highlighted Code Window Line

Say you want to begin emulation from the current program counter location and run until a certain line in your code is reached. This can be done easily in a Code window. To do this, open a source or assembly Code window containing the line you wish to run to. Then highlight the line by clicking on it. When you press the **To Cursor** button, emulation will start and will continue until the highlighted line is reached. Care must be taken, however, to insure that the highlighted line will indeed be reached by your code or emulation will continue unabated.

Single Stepping In A Code Window

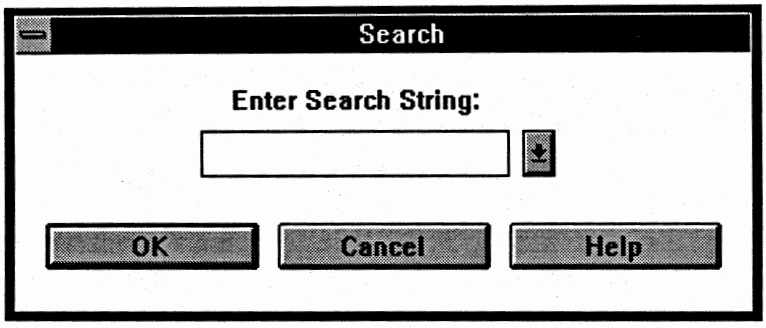
You can perform two different types of single stepping operations while in a Code window, Step Over and Step In.

Pressing the **Step Over** button will begin stepping from the current program counter (PC) location and will step one line of source code if in a source window and will step one line of assembly code if in an assembly window. Using this button **will not** allow the stepping to proceed into any other modules or assembly language routines. The code in these routines is actually executed, but the stepping process does not show this as stepping always appears to be in the active Code window. It should be noted that, if the code being stepped over is large, an hourglass will appear showing you that a step over operation is being performed. This hourglass will not appear if the cursor is pointing to a Code window button.

Pressing the **Step In** button will begin stepping from the current PC location and will step one line of source code if in a source window and will step one line of assembly code if in an assembly window. Using this button **will** allow stepping to proceed into any other module or assembly language routine. When a call or jump occurs in this mode, the source module or assembly rountine is automatically loaded into a Code window in SOURCE or ASSEMBLY mode (respectively), and stepping proceeds in this new window.

Searching For A Code Window String

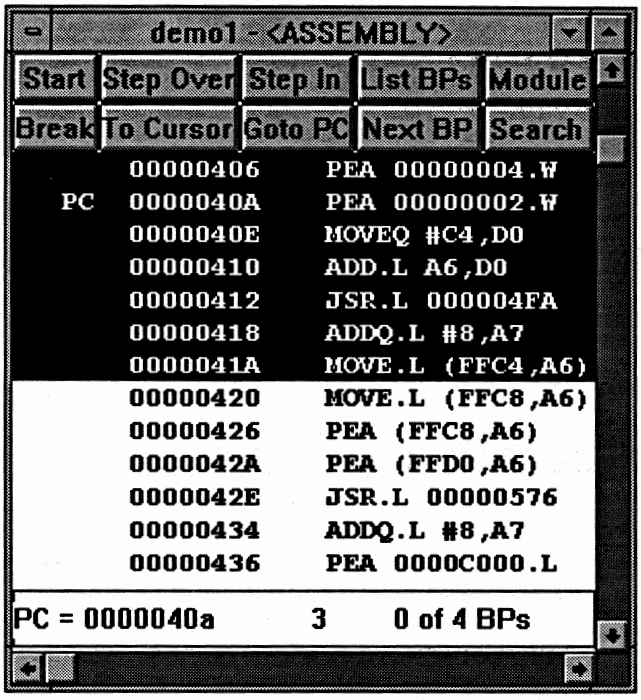
You can search a Code window for a string. This can be useful when looking for a comment or a specific variable name. To begin a search, press the **Search** button. A Search window will be shown as follows:



Enter the string to be searched for and press the **Ok** button to begin the search. Previously entered strings can be recalled by pressing the **Scroll Box** button. Use the **Cancel** button to cancel the search operation. The **Help** button will invoke help for this window.

Viewing Code Window Assembly

If you are working with a Code window that contains source information, you can show the assembly information for that module also. This assembly window will track the source window it was invoked from . The highlighted lines in the assembly window correspond to the highlighted lines in the source window. An example of an assembly window is shown as follows:

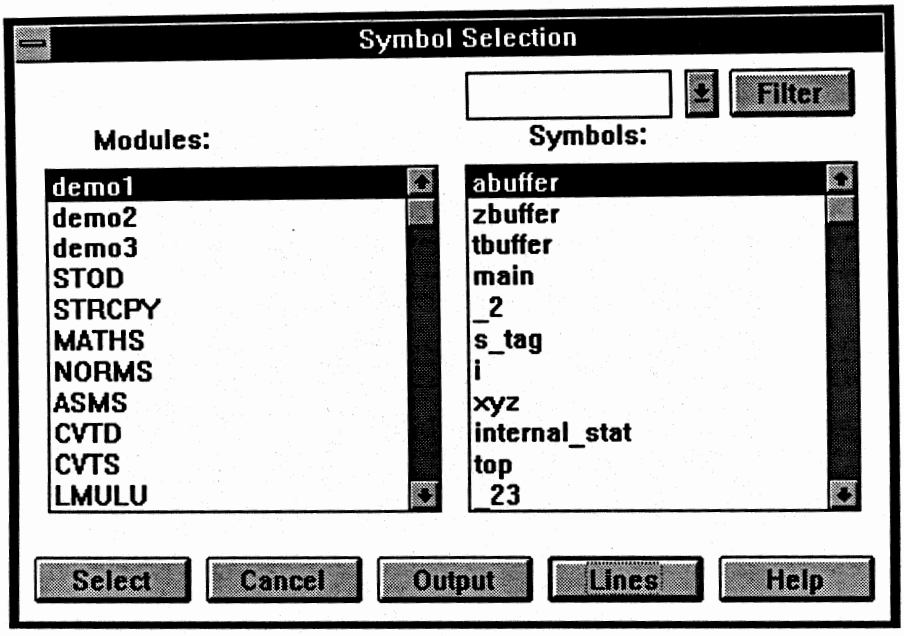


To open the assembly window for a module, you must have a Code window opened that contains a source module. Then, choose 'Source/Assemble' or you can use the accelerator key combination Ctrl+A. The corresponding assembly Code window will be displayed.

Once the assembly window is open, you can make it the active window and use its buttons to perform commands. When you do this, the corresponding source window will then track the assembly window. See the first page of this chapter for an example of a source window and its corresponding assembly window being shown together.

Viewing Code Window Symbols

If you need to see what symbols are defined for a particular module, you can open the Symbol Selection window. An example Symbol Selection window is shown as follows:



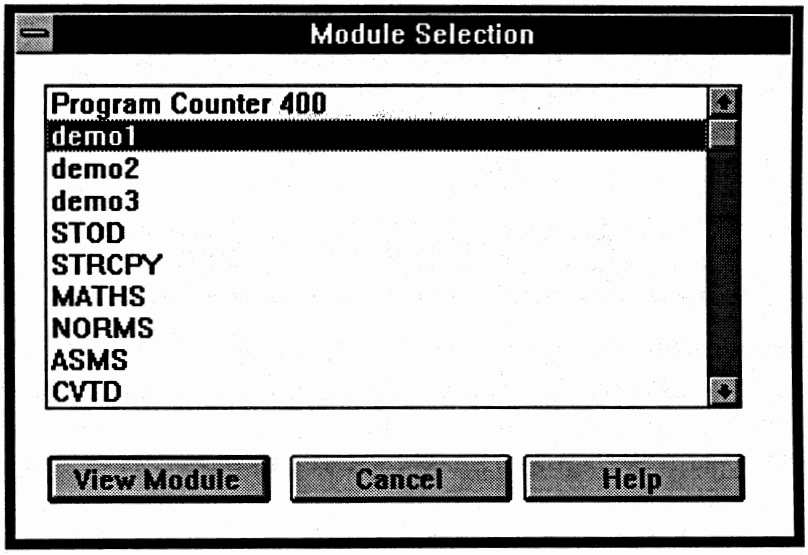
To open this window, choose 'Source/Symbols'. An alternate method of opening the Symbol Selection window is to use the accelerator key combination Ctrl+Y (Windows version only).

The symbols available for a module can by viewed by clicking on the desired module. There is a **Lines** button that will switch the display to the line number symbols available for the module. When line numbers are shown, this button will switch back to **Symbols**. When this window is invoked from a Code window, the **Select** button is not functional.

For additional information on using the Symbol Selection window, see *Working With Symbols* in Chapter 2.

Selecting Other Modules From A Code Window

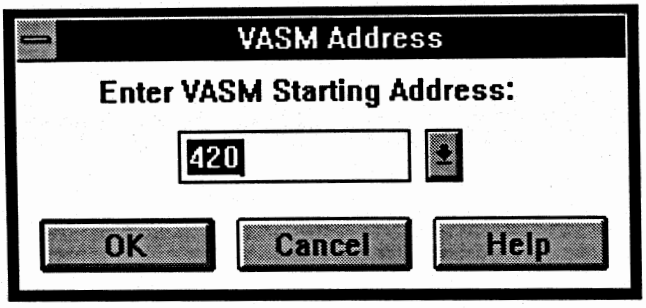
You can select other modules from a Code window by pressing the **Module** button or by using the accelerator key combination Ctrl+O (Windows version only). This will open the Module Selection window which allows you to view the modules that were loaded as part of your download file:



You can now highlight the module you wish to load into a Code window and then press the **View Module** button. A Code window will then be opened which contains the code for the selected module. To cancel the Module Selection operation, press the **Cancel** button. The **Help** button will invoke help for this window.

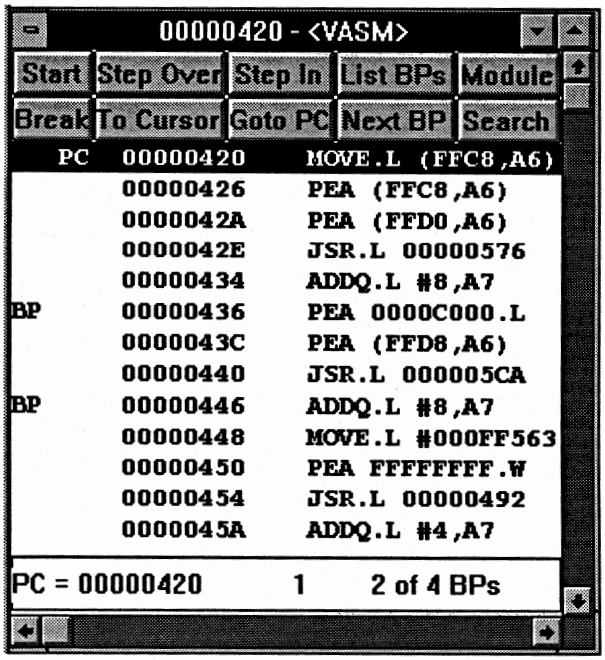
Using The VASM Window

The View Assembly (VASM) window allows you to load a Code window with assembly information beginning at an address you specify. All of the functions of a normal Code window are available in this VASM window. To open a VASM window, choose 'Source/VASM' or use the accelerator key combination Ctrl+V (Windows version only). You will then be prompted with a VASM Address window:



Enter the address from which you wish to start the VASM window in the field provided and then press the **Ok** button. A VASM window will then be opened beginning at this address. Previously entered VASM addresses can be recalled by pressing the **Scroll Box** button. Use the **Cancel** button to cancel the VASM address operation. The **Help** button will invoke help for this window.

An example VASM window that was generated from the above address entry is shown as follows:

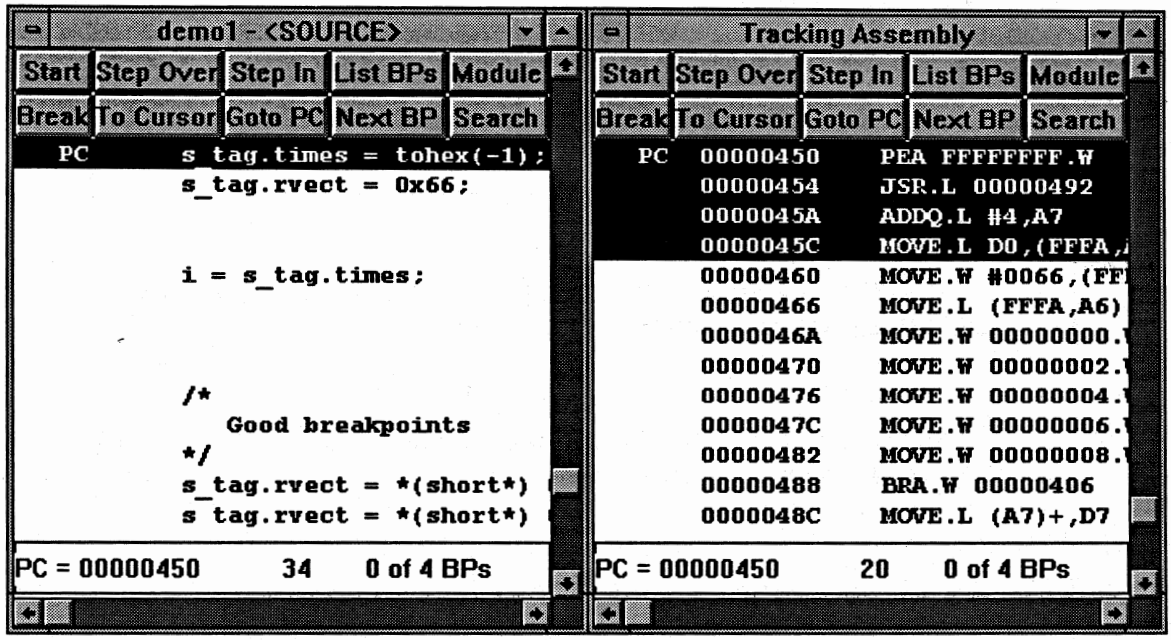


Since this window is just a specialized version of a Code window, see *Using The Code Window* for additional information on using this window.

Using The Tracking Assembly Window

The Tracking Assembly window is an assembly window which will remain on the screen and will track the source window that currently has the focus. This is done by highlighting the assembly lines that correspond to the source line highlighted in the active module. If this window is activated when no source modules are open, it will show assembly code beginning at the current program counter location.

The Tracking Assembly window can be opened by choosing 'Source/Tracking Assembly'. An example Tracking Assembly window,along with the source window that it is tracking, are shown as follows:

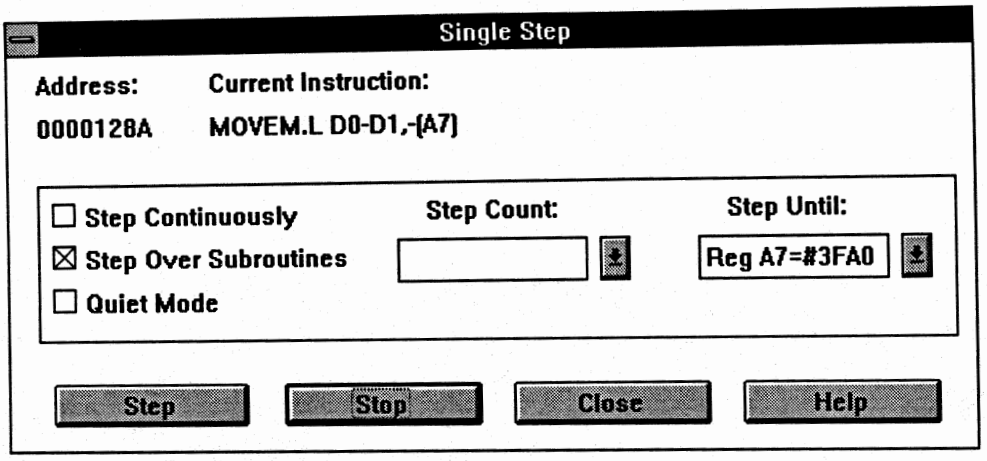


This window can be very useful as you can open it, position it on the screen where you want it, and it will always track whatever source Code window you are currently working in. When you go into assembly, it will mirror the assembly window you are working with.

Since this window is just a specialized version of a Code window, see *Using The Code Window* for additional information on using this window.

Using The Single Step Window

If you prefer to do your single-stepping outside of a Code window, the Single Step window will allow you to do this. This window allows you to single-step at the assembly level. Several conditions define the stepping operation including using step counts and step until conditions to terminate the stepping operation. To open the Single Step window, choose 'Source/Step'. An example Single Step window is shown as follows:



Single Step Window Fields

The Single Step window has several fields that determine how single-stepping occurs. An overview of these fields is given in the sections that follow.

Address Field

The 'Address' field shows the location of the program counter. This is the address that will be executed next when single stepping is performed.

Current Instruction Field

The 'Current Instruction' field shows the instruction pointed to by the program counter. This is the next instruction that will be executed when single stepping is performed.

Step Continuously Check Box

The 'Step Continuously' box, when checked, will allow stepping to proceed continously until the **Stop** button is pressed. If this option is not checked, one instruction will be executed each time the **Step** button is pressed.

Step Over Subroutines Check Box

The 'Step Over Subroutines' box, when checked, will NOT allow stepping to proceed into any called routines. If this option is not checked, stepping will proceed into the called routine.

Quiet Mode Check Box

The 'Quiet' box, when checked, will only show the first instruction executed and will not update the 'Address' and 'Current Instruction' fields until the **Stop** button is pressed. At that point, the last instruction executed will be shown.

Step Count Field

The 'Step Count' field allows you to enter the number of steps desired. When the **Step** button is pressed, the entered number of steps will occur and then stepping will stop. Notice that, when a value is entered in this field, the 'Step Continously' option is disabled. To enter a value in this field, place the mouse cursor in this field and press the left mouse button. The **Scroll Box** button can be used to view and select any previously entered step counts.

Step Until Field

The 'Step Until' field allows you to enter a certain condition on which stepping will stop. When the **Step** button is pressed, stepping will begin and will stop when the specified condition is encountered. To enter a value in this field, place the mouse cursor in this field and press the left mouse button. The **Scroll Box** button can be used to view and select any previously entered step conditions.

The Step Until clause is defined as follows:

```
OPERAND1 OPERATION OPERAND2
```

Where OPERAND1 can be:

```
     Reg <reg> - Register name
Long at <addr> - Long word value at address <addr>
Word at <addr> - Word value at address <addr>
Byte at <addr> - Byte value at address <addr>
```

OPERAND2 can be:

```
 Reg <reg> - Register name
  #<value> - Immediate value (size determined by OPERAND 1)
    <addr> - Value at <addr> (size determined by OPERAND 1)
```

OPERATION can be:

```
 = - OPERAND1 equal to OPERAND2
== - OPERAND1 equal to OPERAND2
!= - OPERAND1 not equal to OPERAND2
<> - OPERAND1 not equal to OPERAND2
>< - OPERAND1 not equal to OPERAND2
 > - OPERAND1 greater than OPERAND2
 < - OPERAND1 less than OPERAND2
>= - OPERAND1 greater than or equal to OPERAND2
=> - OPERAND1 greater than or equal to OPERAND2
<= - OPERAND1 less than or equal to OPERAND2
=< - OPERAND1 less than or equal to OPERAND2
```

Example:

SS Until word at 5000 = #1234

Would stop stepping when 1234H appears at address 5000H. It should be noted that this command takes into account the way data is stored in memory based on the microprocessor manufacturer. If, for example, you wished to stop on a word value of 1234H being in location 5000H, the emulator will look for a 1234H at location 5000H for Motorola processors and 3412H at location 5000H for Intel, Hitachi, and Zilog processors.

Single Step Window Buttons

A description of the functions of the buttons available in this window follows:

Step

The **Step** button will begin stepping if 'Step Continuously' is checked, and will step one instruction each time that the **Step** button is pressed if 'Step Continuously' is not checked.

Stop

The **Stop** button will stop the stepping operation.

Close

The **Close** button will close the Single Step window.

Help

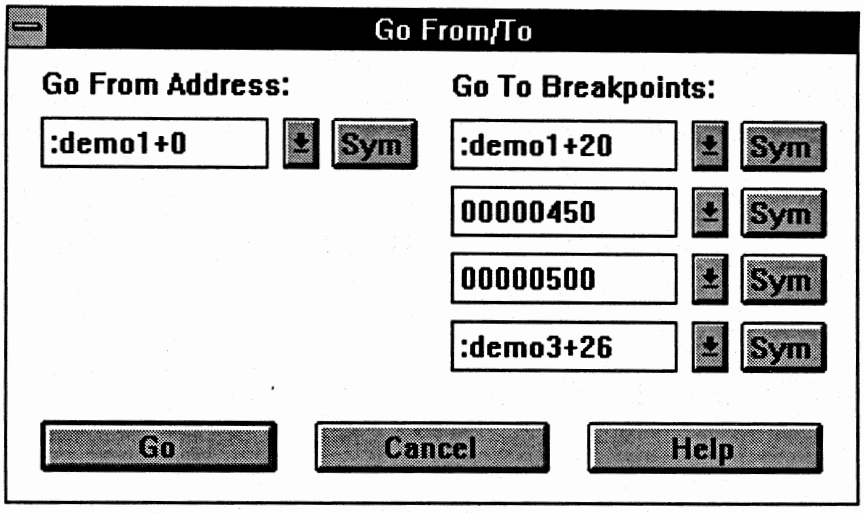
The **Help** button will invoke help for this window.

Scroll Box

Use the **Scroll Box** button to select from previously entered values.

Using The Go From/To Window

The Go From/To window allows you to begin emulation from a specific address and, once emulation is begun, break on up to four defined breakpoints. This can be handy when you need a down and dirty way to set temporary breakpoints without using the command window. To open the Go From/To window, choose 'Source/Go From/To'. An example Go From/To window is shown as follows:



Go From/To Window Fields

The Go From/To window has several fields that determine how this window functions. An overview of these fields is given in the sections that follow.

Go From Address Field

The 'Go From Address' field defines the address or line number where emulation will begin when the **Go** button is pressed. A hexadecimal address can be entered here or a line number of a module can be selected by pressing the **Sym** button. The **Scroll Box** button can be used to select from previously entered values.

Go To Breakpoints Fields

The 'Go To Breakpoints' fields define the addresses or line numbers where emulation will stop when any of them are encountered. The four breakpoints defined here are evaluated as a logical OR condition where emulation will stop when any of these conditions occurs. A hexadecimal address can be entered here or a line number of a module can be selected by pressing the **Sym** button. The **Scroll Box** button can be used to select from previously entered values. **Note: The temporary breakpoint address must reside in RAM since the BMD will attempt to write a BGND instruction there. If the BMD is not able to write to the temporary breakpoint address, the message "Warning: cannot install breakpoint instruction [BGND] at location xxxxxx" will be displayed where "xxxxxx" is the address of the temporary breakpoint.**

Go From/To Window Buttons

A description of the functions of the buttons available in this window follows:

Sym

The **Sym** button will open the Symbol Selection window where you can select a module and line number to be put in the field. **For additional information on using the Symbol Selection window, see *Working With Symbols* in Chapter 2.**

Go

The **Go** button will begin emulation from the defined address. Emulation will stop when the first of the four breakpoint conditions occurs.

Cancel

The **Cancel** button will cancel the defined Go From/To setup and will close the Go From/To window.

Help

The **Help** button will invoke help for this window.

Scroll Box

Use the **Scroll Box** button to select from previously entered values.

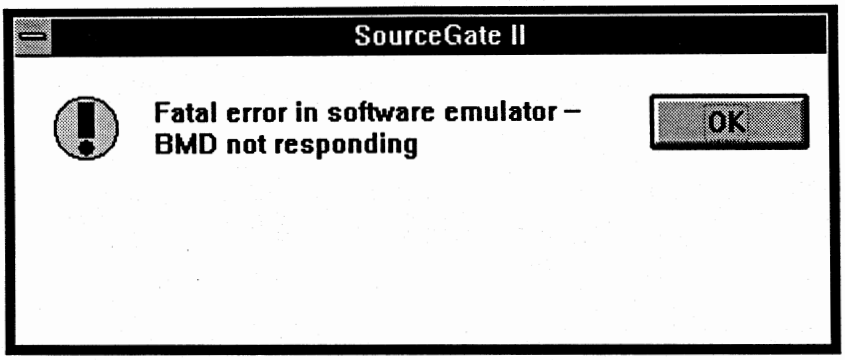
APPENDICES

Appendix A Error Messages 129
Appendix B Using Compilers and Assemblers 134

Appendix A Error Messages

The following is an overview of some of the more common error messages that could be encountered when using the SourceGate II BMD. If you encounter an error message that is not listed here, contact HMI Technical Support.

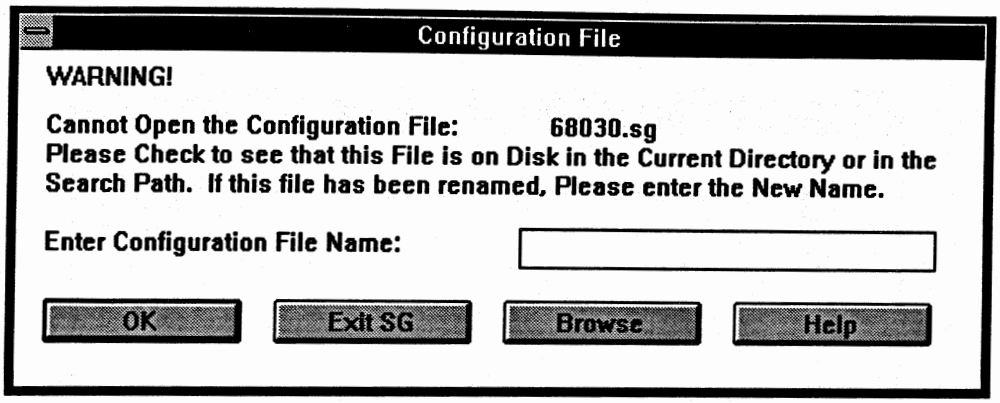
ERROR:



CAUSE:
The SourceGate II BMD could not establish communications with the background mode processor on the target system.

SOLUTION:
This problem can be caused by several conditions. First, make sure that the target system is powered on. Second, make sure that a straight-through type of RS-232 cable is plugged into the HMI supplied cable on one end and to a valid serial port on the computer end. Third, the Berg connector on the other end of the HMI supplied cable must be plugged into the target system correctly. Finally, make sure that the SourceGate II BMD is being invoked with the proper COM port setting as outlined in the section ***Starting The SourceGate II BMD***. If you still have problems after trying the above fixes, call Technical Support.

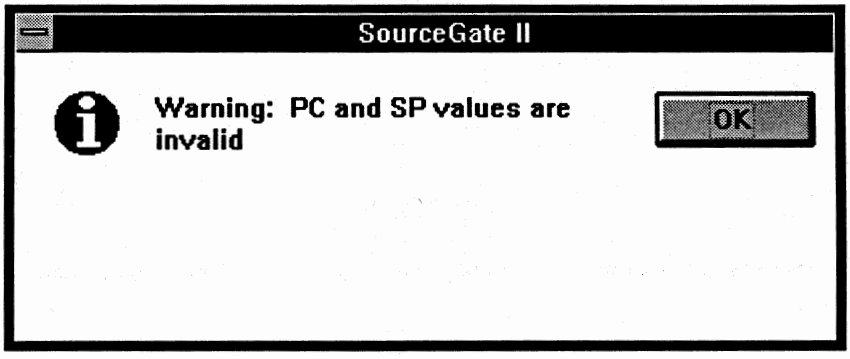
ERROR:


Configuration File
WARNING!
Cannot Open the Configuration File: 68030.sg
Please Check to see that this File is on Disk in the Current Directory or in the Search Path. If this file has been renamed, Please enter the New Name.
Enter Configuration File Name:
OK
Exit SG
Browse
Help

CAUSE:
The personality file for your particular BMD could not be found in the current directory or in the search path.

SOLUTION:
Use the **Browse** button to find the correct drive/directory and .SG filename. You can also move the named file to your current directory, change to the directory where this file resides, or include the directory where this file resides in your search path. Then, press the **OK** button.

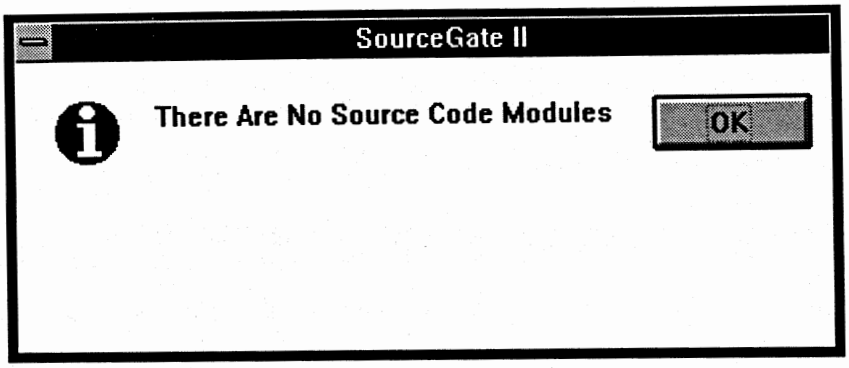
ERROR:



CAUSE:
Certain processors, most notably the 68000, 68020, 68030, 68040, 68302, 68306, and 68300 family, fetch their reset and stack pointer vectors from memory upon power-up. If this memory is mapped to uninitialized RAM, the values fetched for these vectors may be random garbage until they are initialized with a valid value.

SOLUTION:
Edit memory locations 0 - 7 so that they contain valid reset and stack pointer vectors and then perform the "RS" command from the Command window, or press the "RESET" button in the Run Status window.

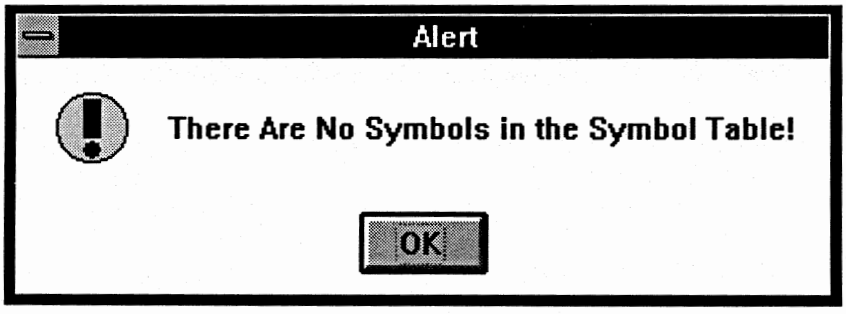
ERROR:



CAUSE:
You attempted to perform a source-level debugging operation. However, a file containing source-level debugging information has not yet been loaded.

SOLUTION:
Load a file that contains source-level debugging information and try again.

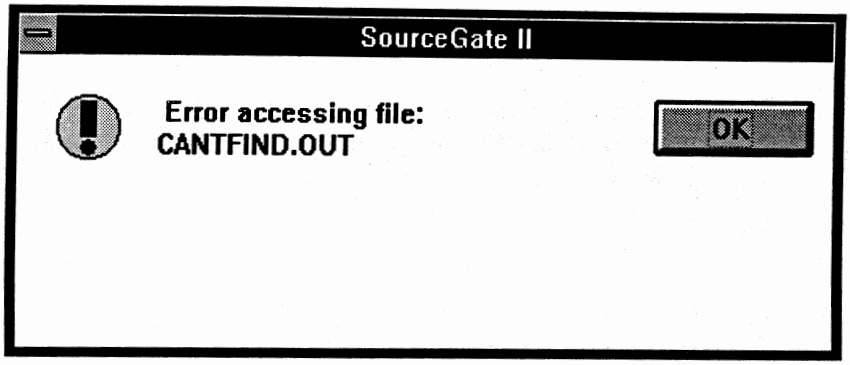
ERROR:



CAUSE:
You attempted to perform an operation that referenced a symbol. However, a file containing symbolic debugging information has not yet been loaded.

SOLUTION:
Load a file that contains symbolic debugging information and try again.

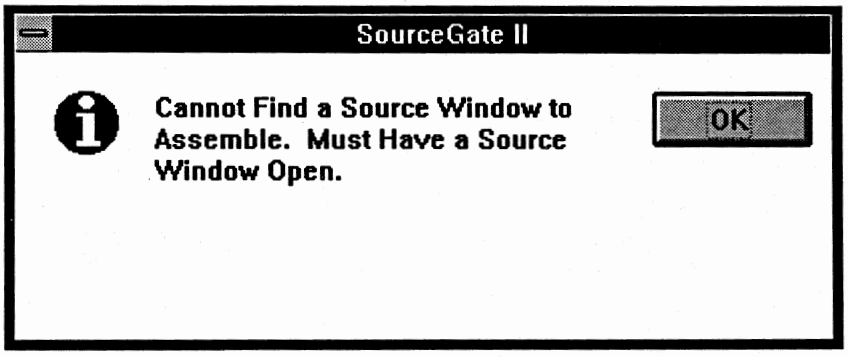
ERROR:



CAUSE:
The SourceGate II BMD cannot find the file specified.

SOLUTION:
Make sure that the filename and path specified are correct.

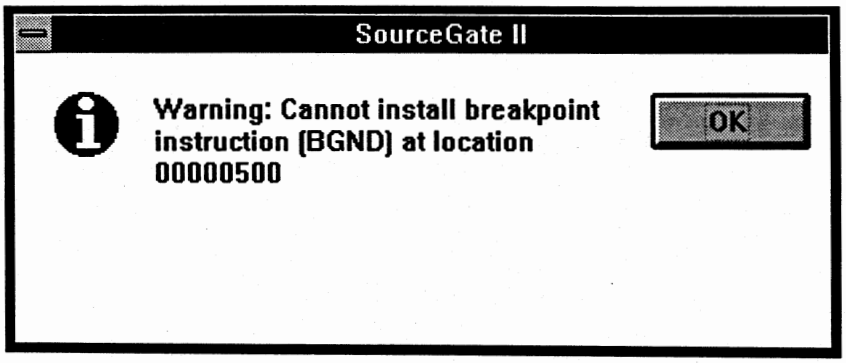
ERROR:



CAUSE:
You tried to invoke an assembly window when no source modules are active.

SOLUTION:
Open a source window and then invoke an assembly window, or use the VASM command to open an assembly window. The VASM command allows you to open an assembly window at any time beginning at the defined address.

ERROR:



CAUSE:
The SourceGate II BMD cannot write a BGND instruction to the address shown. Any time a breakpoint is set in the BMD, it must be set at a location that contains RAM so that the BGND instruction can be written at that location.

SOLUTION:
Make sure that the address that your are setting the breakpoint on resides in RAM.

Appendix B Using Compilers and Assemblers

SourceGate is compatible with a number of downloadable file formats. SourceGate can load the code, data, symbolic, and source-level debugging information (line numbers) from those formats that supply such information.

SourceGate supports the following formats (those that do not produce symbol information (including line numbers) are so noted):

Motorola S-Record (No symbols)
Intel Hex (No symbols)
Tekhex (No symbols)
HMI Binary (No symbols)
Common Object File Format (COFF)
Software Development Systems (SDS)
Archimedes Debug (IAR)
IEEE-695
Extended Tekhex
Microtec Research Absolute
Hitachi

Most compiler/assembler manufacturers provide linkers that can produce one of the above formats. Other formats can be converted to Extended Tekhex with object file convertors supplied by HMI. These converters are available through the HMI-BBS at 205-881-7395, 8-N-1, 24 hours a day, 7 days a week. If you don't have access to a modem, call us and we'll send you a converters disk. All HMI converters supplied on the converters disk are outlined in the following paragraphs.

The HMI converters disk contains executable (.EXE or .COM) and documentation (.DOC) files for each converter listed in the following sections. Please read the applicable documentation file for the converter you wish to use **before** running the converter. The documentation files have been written to provide insight on how to use the converter programs properly.

NOTE: Chances are you will not need any of these converters in order to read your object files correctly. If you have a doubt about whether or not your output format is supported directly, try reading the file directly into the SourceGate II BMD. If it is a valid format, it will be read correctly. If not, you will get an error message. You can then use one of the converters to convert the file into one of the above formats. If you have any questions about what type of output file your compiler/assembler produces, call your software supplier or HMI Technical Support.

INDEX TO THE FILE CONVERTERS DISK

180AD2XT	-	2500 A.D. 64180 symbol table file to XTEKHEX file converter.
2500TOXT	-	2500 A.D. standard symbol table to Extended TEKHEX file converter.
68KAD2XT	-	2500 A.D. MC68000 symbol table file to Extended TEKHEX file converter
AATOXT	-	American Automation hex file to Extended TEKHEX file converter.
ATOXT	-	UNIX A.OUT file to Extended TEKHEX file converter.
AVOXTEK	-	AVOCET plain dump symbol file to Extended TEKHEX file converter.
DBGXTEK	-	Manx debug (DBG) file to Extended TEKHEX file converter.
HTOXT	-	HITACHI hex file to Extended TEKHEX file converter.
ITOB	-	INTEL hex file to BINARY file converter.
ITOM	-	INTEL hex file to MOTOROLA 'S' record file converter.
MTTOXT	-	MICROTEK symbol table to Extended TEKHEX file converter.
OBJCNV	-	INTEL Object file converter. (The Intel Object file converter operates on an Intel object format file produced by the Intel programs LOCATE, RL51, and/or RL96. It produces as its output a standard Intel format HEX file, and a standard HMI format symbol file.)
QUEXTEK	-	QUELO linker report file to Extended TEKHEX file converter.
UTILXTEK	-	Intermetrics (Whitesmiths) Object file to Extended TEKHEX file converter.
ZTOXT	-	INTEL hex file/ZILOG symbol file to Extended TEKHEX file converter.

INDEX TO THE FILE CONVERTERS DISK (Cont.)

IAR2XT	-	IAR (Archemedies) UBROF Debug file to Extended TEKHEX file converter.
COFF2XT	-	COFF file (Introl, Sierra, Intermetrics, SUN386, and PG52x0 COFF) to Extended TEKHEX file converter.
IEEETOXT	-	IEEE 695 Object format file to Extended TEKHEX file converter.
IEEE2IAR	-	IEEE 695 Object format file to Archimedes Debug (IAR) file converter.
NWISTOXT	-	Northwest Instruments Symbol file to Extended TEKHEX file converter.
COMFORXT	-	COMFOR file to Extended TEKHEX file converter.
ARCH2BNK	-	Archimedes Banked "S" Record Splitter
OS92HMI	-	Microware OS2 Format To SDS Format File Converter

Index

SYMBOLS

:, 37
:=A, 38
:=C, 38
:=D, 39
:=M, 39
:=S, 39
:module/symbol, 37
:module/symbol=value, 37
:symbol, 37
:symbol*, 37
:symbol=value, 37

A

A (In-Line Assembly), 42
Accelerator Keys, 27
Adding A Watch Variable, 83
Assembly Mode Command, 38
Assembly Window, 12
Automating The SourceGate II BMD, 26, 27

B

Begin Emulation, 12
BM (Byte Mode), 43

C

C (Compare Memory Block), 44
Checking Your Package, 1
Clear Symbol Table Command, 38
CM (Show Status Of Control Board Memory), 46
Code Window, 110
Code Window Format, 112
Command File, 10
Command file, 10
Command Line Options, 5
Command Window, 11, 35
Comparing Memory Blocks, 96
Control Menu Boxes, 9

D

D (Dump Memory Block), 47
Deleting A Watch Variable, 89
Deleting Code Window Breakpoints, 113
Disk Space Requirements, 2
Display Current Modes Command, 39
DM (Data Memory Mode), 48
Download Hex File, 29
Download Hex File, 10
Download Options, 30

E

E/EN (Enter/Change Memory), 49
Environment Variables, 7
Error Messages, 129
Exiting SourceGate II, 11
Exiting The SourceGate II BMD, 25
EXP (Examine/Change Peripheral), 50
EXR (Examine/Change Peripheral Register), 51

F

F (Fill Memory Block), 52
File Converters Disk, 135
Filling Memory With A Constant, 98
Finding Subsequent Code Window Breakpoints, 116
Finding The Current Program Counter While In A Code Window, 116

G

G (Begin Real-Time Emulation), 53
Go From/To Window, 12, 126

H

Help System, 13, 19
HLD (Hex Listing Disabled), 54
HLE (Hex Listing Enabled), 55

I

I/IN (Input), 56
Iconizing A Window, 15
Installing The Motif Version Of The SourceGate II BMD, 3
Installing The SUN Version Of The SourceGate II BMD, 2
Installing The Windows Version Of The SourceGate II BMD, 2

L

L (List Disassembled Code), 57
Listing Code Window Breakpoints, 115
LM (Long Word Display Mode), 58
Log Input/Output To File, 11

M

M (Move Memory Block), 59
Maximize Button, 10
Memory Compare Window, 96, 98
Memory Goto Window, 99
Memory I/O Window, 100
Memory Move Window, 101
Memory Requirements, 2
Memory Search Window, 102
Memory Window, 11, 95
Minimize Button, 10
Mixed Mode Command, 39
Modifying Memory Values, 96
Modifying Register Values, 94
Module Commands, 38
Module Selection Window, 12, 111, 120
Moving Memory Blocks, 101

O

O/ON (Output), 60
Object file, 10
Object File Formats, 29
Opening A Code Window, 111

P

PM (Program Memory Mode), 61
Printer Setup, 11
PROM, 10, 33

R

RD (Read Object File Into Memory), 62
Reading A Specific Memory Location, 100
Register Window, 11, 93
Reset Emulator, 12
Restore Button, 10
Restoring A Saved Watch Window, 89
RP (Reset Processor, Continue Emulation), 64
RS (Reset Processor, Stop Emulation), 65
Run Status Window, 11, 108
Run Status Window Fields, 108
Running To A Highlighted Code

Index

Window Line, 116

S

Saving A Watch Window, 89
Scroll Bars (Horizontal and Vertical), 10
Scroll Boxes (Horizontal and Vertical), 10
Searching For A Code Window String, 117
Searching Memory Blocks, 102
Selecting A Printer, 18
Selecting Other Modules From A Code Window, 120
Setting Code Window Breakpoints, 113
Single Step Window, 12, 123
Single Stepping In A Code Window, 116
SM (Supervisor Memory Mode), 66
Source Code Interface, 110
Source Code Interface Options, 13, 28
Source Mode Command, 39
SourceGate II BMD Menu Bar, 9, 10
SourceGate II BMDTitle Bar, 10
SP (Stop Emulation), 67
SR/SRN (Search Block Of Memory), 68
SS (Single Step), 70
ST (Show Emulator Status), 73
Stack Trace Window, 11, 106
Starting The Motif Version Of The SourceGate II BMD, 5
Starting The SourceGate II BMD From File Manager, 4
Starting The SourceGate II BMD From Program Manager, 4
Starting The SUN Version Of The SourceGate II BMD, 4
Starting The Windows Version Of The SourceGate II BMD, 4
Stop Emulation, 12
Symbol Commands, 37
Symbol Selection Window, 12, 19, 88, 119
System Requirements, 1

T

Tracking Assembly Window, 12, 122

U

UM (User Memory Mode), 74
Upload Hex File, 10, 33
Using Compilers and Assemblers, 134

V

VASM (View Assembly), 75
VASM Window, 12, 121
VER (Display Firmware Version), 76
Viewing A Specific Address In Memory, 99
Viewing Code Window Assembly, 118
Viewing Code Window Symbols, 119

W

W (Write Memory Contents To File), 77
Watch Window, 11, 82
Watch Window Examples, 90
Window Manipulation, 13, 17
WM (Word Display Mode), 79
Writing A Specific Memory Location, 100

X

X (Examine/Change Registers), 80